Journal of Artificial Intelligence Research 68 (2020) 1-99 Submitted 01/20; published 07/20

Towards Partial Order Reductions for Strategic Ability

Wojciech Jamroga W.JAMROGA@IPIPAN.WAW.PL
Institute of Computer Science,
Polish Academy of Sciences, Warsaw, Poland
Interdisciplinary Centre for Security, Reliability, and Trust,
University of Luxembourg, Luxembourg

Wojciech Penczek PENCZEK@IPIPAN.WAW.PL
Institute of Computer Science,
Polish Academy of Sciences, Warsaw, Poland

Teofil Sidoruk T.SIDORUK@IPIPAN.WAW.PL
Institute of Computer Science,
Polish Academy of Sciences, Warsaw, Poland
Faculty of Mathematics and Information Science,
Warsaw University of Technology, Warsaw, Poland

Piotr Dembiński PIOTRD@IPIPAN.WAW.PL
Antoni Mazurkiewicz AMAZ@IPIPAN.WAW.PL
Institute of Computer Science,
Polish Academy of Sciences, Warsaw, Poland

Abstract

We propose a general semantics for strategic abilities of agents in asynchronous systems, with and without perfect information. Based on the semantics, we show some general complexity results for verification of strategic abilities in asynchronous interaction. More importantly, we develop a methodology for *partial order reduction* in verification of agents with imperfect information. We show that the reduction preserves an important subset of strategic properties, with as well as without the fairness assumption. We also demonstrate the effectiveness of the reduction on a number of benchmarks. Interestingly, the reduction does not work for strategic abilities under perfect information.

1. Introduction

Multi-agent systems describe interactions of multiple entities called *agents*, often assumed to be intelligent and autonomous. *Alternating-time temporal logic* **ATL*** and its fragment **ATL** (Alur et al., 1997, 2002) extend temporal logic with the game-theoretic notion of *strategic ability*. They allow to express statements about what agents (or groups of agents) can achieve. For example, $\langle\langle i \rangle\rangle \mathrm{F}\, \mathsf{win}_i$ says that agent i has the ability to eventually win no matter what the other agents do, while $\langle\langle i, j \rangle\rangle \mathrm{G}\, \mathsf{safe}$ expresses that agents i and j 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. Moreover, algorithms and tools for verification of strategic abilities have been in constant development

for almost 20 years (Alur et al., 1998, 2001; Kacprzak & Penczek, 2004; Lomuscio & Raimondi, 2006; Chen et al., 2013; Huang & van der Meyden, 2014; Busard et al., 2014; Pilecki et al., 2014; Cermák et al., 2014; Lomuscio et al., 2017; Cermák et al., 2015; Belardinelli et al., 2017a, 2017c).

However, there are two caveats. First, many tools and algorithmic solutions focus on agents with perfect information, i.e., agents who always know exactly the global state of the system. This is clearly unrealistic in all but the simplest multi-agent scenarios. Still, the tendency is somewhat easy to understand, since model checking of **ATL** variants with *imperfect* information is $\mathbf{\Delta_2^P}$- to **PSPACE**-complete for agents playing memoryless strategies (Schobbens, 2004; Jamroga & Dix, 2006; Bulling et al., 2010) and **EXPTIME**-complete to undecidable for agents with perfect recall of the past (Dima & Tiplea, 2011; Guelev et al., 2011; Berthon et al., 2017b). Moreover, the imperfect information semantics of **ATL** does not admit alternation-free fixpoint characterizations (Bulling & Jamroga, 2011; Dima et al., 2014, 2015), 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 (Lomuscio & Raimondi, 2006; Raimondi, 2006; Lomuscio et al., 2009, 2017), but the issue has never been at the heart of the tool. Indeed, the recent attempts at practical model checking of imperfect information strategies (Pilecki et al., 2014; Busard et al., 2014; Huang & van der Meyden, 2014; Busard et al., 2015; Jamroga et al., 2019; Kurpiewski et al., 2019a, 2019b) confirm that the problem 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 or non-negligible delays in execution of actions. No less importantly, many systems that are synchronous at the implementation level (say, the level of the virtual machine) can be more conveniently modeled as asynchronous on a more abstract level. 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 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 an 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. First, we define a semantics of strategic abilities for agents in asynchronous systems, with and without perfect information. Secondly, we present some general complexity results for verification of strategic abilities in such systems. Thirdly,

and most importantly, we adapt *partial order reduction (POR)* to model checking of strategic abilities for agents with imperfect information. We also present experimental results demonstrating that POR allows to significantly reduce the size of the model, and thus to make the verification more feasible. In fact, we show that the most efficient variant of POR, defined for linear time logic **LTL**, can be applied almost directly. The (nontrivial) proof that the **LTL** reductions work also for the more expressive strategic operators is the main contribution of this paper. Interestingly, the scheme does *not* work for verification of agents with perfect information.

The outline of the paper is as follows. In Section 2, we introduce the structures to represent and reason about asynchronous multi-agent systems. In Section 3, we define the semantics of **ATL** for asynchronous systems. In Section 4, we show the general complexity results. Sections 5 and 6 put forward the theoretical foundations and the algorithms for partial order reduction. Experimental results for several multi-agent scenarios are presented in Section 7. We conclude in Section 8.

Related work. Relevant related work is relatively scarce. Asynchronous semantics and partial order reduction for distributed systems were extensively studied by Peled (1993, 1996a, 1996b, 1998), Kokkarinen et al. (1997), Godefroid and Wolper (1994), Godefroid (1991), Gerth et al. (1999), Kristensen and Valmari (2000), Penczek et al. (2000). These approaches deal with reductions for preserving reachability and temporal logics: **LTL**, **CTL**, and **ACTL**. The only efficient approach to partial order reduction in a MAS context (Lomuscio et al., 2010a, 2010b) concerns standard temporal-epistemic logics ($\mathbf{LTLK_{-X}}$, $\mathbf{CTL^*K_{-X}}$) interpreted over interleaved interpreted systems. Our approach for **ATL** is similar to the approach for **LTL**, but the logic is interpreted over models based on interleaved interpreted systems like in case of $\mathbf{LTLK_{-X}}$. As shown by Meski et al. (2014) interleaved interpreted systems can be viewed as a subclass of interpreted systems e.g., by adding dummy epsilon actions. The most recent approaches include dynamic POR (Flanagan & Godefroid, 2005; Abdulla et al., 2014; Chatterjee et al., 2016) and combine POR with symbolic methods (Kahlon et al., 2009; Konnov et al., 2015). Our future plan is to extend our POR approach for **ATL** to dynamic and symbolic versions.

Alur, Henzinger and Kupferman mentioned asynchronous systems in their seminal paper on **ATL** (Alur et al., 2002), but they modeled them as a special case of synchronous systems. Asynchronous omega-regular games were also considered by Puchala (2010). Reactive modules (Alur & Henzinger, 1999), the class of representations behind the Mocha model checker (Alur et al., 1998, 2001), feature several modes of asynchronous execution, but – to the best of our knowledge – this aspect has never been given a more systematic analysis. The work that comes closest to our new proposal is (Dastani & Jamroga, 2010) where a variant of **ATL** was proposed for agent-oriented agent programs written in 2APL with asynchronous execution semantics.

2. Models of Multi-agent Systems

We first define models of asynchronous interaction in MAS, inspired by Priese (1983), Fagin et al. (1995), Lomuscio et al. (2010a).

2.1 Asynchronous Multi-agent Systems

In many multi-agent systems, the interaction between different agents is asynchronous. That is, the actions of different agents are usually executed independently, without clear temporal relationship. There is no global clock that provides automatic synchronization to all the components in the system. Only some events, such as synchronous communication, require that the involved components execute their parts of the event at precisely the same moment (or within the same time interval).

One can represent such systems with *networks of automata* that execute asynchronously by interleaving local transitions, and synchronize their moves whenever a shared event occurs. This modeling approach is standard in *theory of concurrent systems*, where the use of automata networks dates back at least to the early 1980s and the idea of APA Nets (asynchronous, parallel automata nets) (Priese, 1983). The idea is to represent the behaviour of each component by a finite automaton where the states of the automaton correspond to the local states of the component. The transitions in the automaton are labeled by the events in which the component can take part. Then, the global behaviour of the system is obtained by the interleaving of local transitions, assuming that, in order for an event to occur, all the corresponding components must execute it in their automatons.

We note that the same idea has been also used in the multi-agent systems community. While most models of MAS, such as concurrent game structures (Alur et al., 2002), assume synchronous execution of actions from all the agents, we note that asynchronous execution based on interleaving was considered in some seminal works on agent interaction, both theoretical (Fagin et al., 1995; Lomuscio et al., 2010a) and practically oriented (Alur & Henzinger, 1999). Moreover, building models of MAS from local state spaces is standard in most approaches to practical modeling and verification of MAS (Lomuscio & Sergot, 2003; van der Hoek et al., 2006; Jamroga & Agotnes, 2007), including the most popular model checking tool MCMAS (Lomuscio & Raimondi, 2006; Lomuscio et al., 2017). This motivates the following definition.

Definition 1 (Asynchronous MAS)**.** *An* asynchronous multi-agent system (AMAS) *consists of n agents $\mathcal{A} = \{1, \ldots, n\}$,*[1] *each associated with a tuple $A_i = (L_i, \iota_i, Evt_i, P_i, T_i, \mathcal{PV}_i, V_i)$ that includes a set of* possible local states $L_i = \{l_i^1, l_i^2, \ldots, l_i^{n_i}\}$*, an* initial state $\iota_i \in L_i$*, and a set of* events $Evt_i = \{\alpha_i^1, \alpha_i^2, \ldots, \alpha_i^{m_i}\}$ *in which agent i can choose to participate. Note that the sets Evt_i do not need to be disjoint, i.e., there may be events that require participation of more than one agent.*

$Evt = \bigcup_{i \in \mathcal{A}} Evt_i$ is the set of all events, and $Loc = \bigcup_{i \in \mathcal{A}} L_i$ is the set of all local states in the system. For each event $\alpha \in Evt$, the set $Agent(\alpha) = \{i \in \mathcal{A} \mid \alpha \in Evt_i\}$ contains the agents which have α in their sets of events.

A local protocol $P_i : L_i \to 2^{Evt_i}$ *shows which events are available for selection at which local state. Moreover, $T_i : L_i \times Evt_i \rightharpoonup L_i$ is a (partial)* local transition function *such that $T_i(l_i, \alpha)$ is defined iff $\alpha \in P_i(l_i)$. That is, $T_i(l, \alpha)$ indicates the result of executing event α in local state l from the perspective of agent i.*

Finally, we assume that each agent i in the AMAS is endowed with a set of its local propositions $\mathcal{PV}_i$*, and their valuation $V_i : L_i \to 2^{\mathcal{PV}_i}$. Additionally, the overall set of propositions $\mathcal{PV} = \bigcup_{i \in \mathcal{A}} \mathcal{PV}_i$ collects all the local propositions.*

1. We do not consider the environment component, which may be added with no technical difficulty.

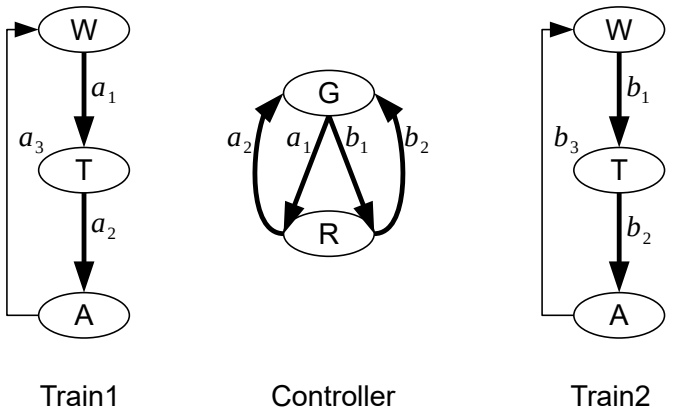

Figure 1: Asynchronous MAS for the TGC benchmark.

Example 1 (TGC)**.** *Figure 1 presents the Train-Gate-Controller (TGC) benchmark (Alur et al., 1993, 1998; Hoek & Wooldridge, 2002). The system consists of three agents: a controller c and two trains t_1, t_2. The trains run on separate circular tracks that jointly pass through a narrow tunnel. Each train can be waiting for the permission to enter (state W), riding inside the tunnel (T), or riding somewhere away of the tunnel (A). The controller switches between green light (state G) and red light (R). Initially, both trains are waiting and the controller displays Green.*

2.2 Interleaved Interpreted Systems

To understand the interaction between asynchronous agents, we use the standard execution semantics from concurrency models, i.e., interleaving with synchronization on shared events. To this end, we unfold the network of local automata (i.e., AMAS) to a single automaton based on the notions of *global states* and *global transitions*, defined formally below.

Definition 2 (Interleaved Interpreted System)**.** *Let $\mathcal{PV}$ be a set of propositional variables. An* interleaved interpreted system (IIS)*, or a* model*, is an asynchronous MAS extended with the following elements: a set $St \subseteq L_1 \times \cdots \times L_n$ of global states, an initial state $\iota \in St$, a partial global transition function $T : St \times Evt \rightharpoonup St$, and a valuation of propositions $V : St \to 2^{\mathcal{PV}}$. For state $g = (l_1, \ldots, l_n)$, we denote the local component of agent i by $g^i = l_i$. Also, we will sometimes write $g_1 \xrightarrow{\alpha} g_2$ instead of $T(g_1, \alpha) = g_2$.*

We will show in Definition 3 how to generate such a model for a given asynchronous multi-agent system.

We say that event $\alpha \in Evt$ is *enabled* at $g \in St$ if $g \xrightarrow{\alpha} g'$ for some $g' \in St$. The global transition function is assumed to be serial, i.e., at each $g \in St$ there exists at least one enabled event.

An infinite sequence of global states and events $\pi = g_0\alpha_0 g_1\alpha_1 g_2 \ldots$ is called an (interleaved) *path* if there is a sequence of global transitions from g_0 onwards, i.e., if $g_i \xrightarrow{\alpha_i} g_{i+1}$ for every $i \geq 0$. $Evt(\pi) = \alpha_0\alpha_1\alpha_2 \ldots$ is the sequence of events in π, and $\pi[i] = g_i$ is the i-th global state of π. $\Pi_M(g)$ denotes the set of all paths in an IIS M starting at g.

IIS can be used to provide an execution semantics to AMAS.

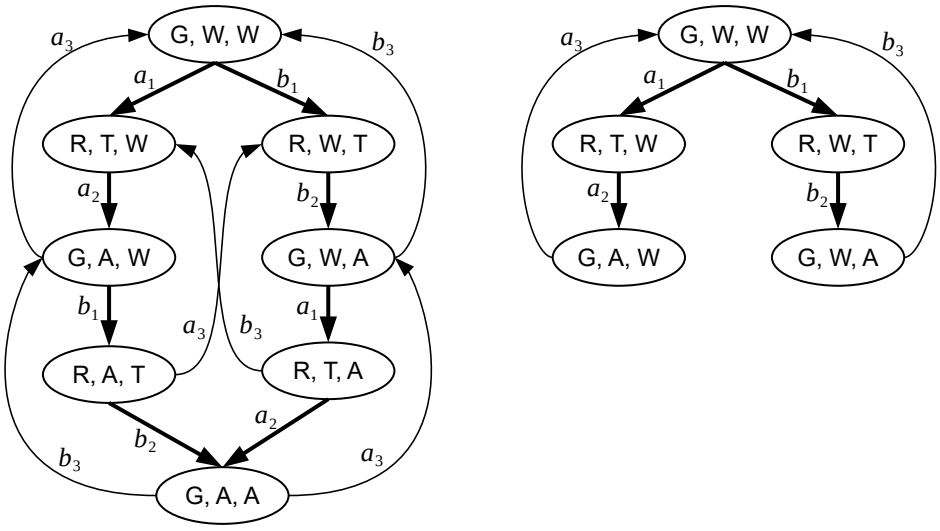

Figure 2: IIS for TGC: full model (left) and reduced model (right). Visible transitions are depicted by bold arrows.

Definition 3 (Canonical IIS)**.** *Let S be an asynchronous MAS with n agents. Its* canonical model *$IIS_V(S)$ extends S with global states $St = L_1 \times \ldots \times L_n$, initial state $\iota = (\iota_1, \ldots, \iota_n)$, and transition function T defined as follows: $T(g_1, \alpha) = g_2$ iff $T_i(g_1^i, \alpha) = g_2^i$ for all $i \in Agent(\alpha)$, and $g_1^i = g_2^i$ for all $i \in \mathcal{A} \setminus Agent(\alpha)$. Moreover, the global valuation of propositions is defined as $V(l_1, \ldots, l_n) = V_i(l_i)$.*

Intuitively, the global states in $IIS_V(S)$ can be seen as the possible configurations of local states of all the agents. Moreover, the transitions are labeled by events that can be synchronously selected (in the current configuration) by all the agents that have the event in their repertoire. Clearly, private events (i.e., events such that $Agent(\alpha)$ is a singleton) require no synchronization.

We also note that, for the properties considered in this paper, the global states unreachable from ι can be as well omitted from the set St.

Example 2. *Let TGC_n be the asynchronous MAS consisting of the controller c and n trains $(t_1, \ldots, t_n)$. Additionally, let us assume $\mathcal{PV} = \{\mathsf{in_1}, \ldots, \mathsf{in_n}\}$ with $\mathsf{in_i} \in V(g)$ iff $g^i = T$. That is, proposition $\mathsf{in_i}$ denotes that train t_i is currently in the tunnel. The state/transition structure of the canonical interleaved interpreted system for TGC_2 is depicted in Figure 2 (left).*

It is easy to see that the global state space grows exponentially with the number of agents. In some cases, it suffices to consider a subset of states and transitions, i.e., concentrate on a submodel of $IIS(S)$.

Definition 4 (Submodel)**.** *Let M, M' be two models extending the same AMAS, such that $St' \subseteq St$, $\iota \in St'$, T is an extension of T', and $V' = V|_{St'}$. Then, we write $M' \subseteq M$ and call M' a* submodel *of M or a* reduced model *of M.*

An example submodel of the IIS for TGC is shown in Figure 2 (right). Note that, for each $g \in St'$, we have $\Pi_{M'}(g) \subseteq \Pi_M(g)$.

In order to generate reduced models, we need a notion of *invisibility* and *independency* of events with respect to propositional variables. Intuitively, an event is invisible iff it does not change the valuations of the propositions. Note that this concept of invisibility is technical, and is not connected to the view of any agent in the sense of Malvone et al. (2017). Additionally, we can designate a subset of agents A whose events are visible by definition. Furthermore, two events are weakly independent iff they are not events of the same agent, and strongly independent iff they are weakly independent and at least one of them is invisible.

Definition 5 (Invisible events)**.** *Consider a model M, a subset of agents $A \subseteq \mathcal{A}$, and a subset of propositions $PV \subseteq \mathcal{PV}$. An event $\alpha \in Evt$ is* invisible *w.r.t. A and PV if $Agent(\alpha) \cap A = \emptyset$ and for each two global states $g, g' \in St$ we have that $g \xrightarrow{\alpha} g'$ implies $V(g) \cap PV = V(g') \cap PV$. The set of all invisible events for A, PV is denoted by $Invis_{A,PV}$, and its closure – of visible events – by $Vis_{A,PV} = Evt \setminus Invis_{A,PV}$.*

Definition 6 (Independent events)**.** *Two notions of independence of events are defined:*

> Weak independence $WI \subseteq Evt \times Evt$ *is defined as:* $WI = \{(\alpha, \alpha') \in Evt \times Evt \mid Agent(\alpha) \cap Agent(\alpha') = \emptyset\}$.
>
> Strong independence *(or simply* independence*)* $I_{A,PV} \subseteq Evt \times Evt$ *is defined as:* $I_{A,PV} = WI \setminus (Vis_{A,PV} \times Vis_{A,PV})$. *Two events $\alpha, \alpha' \in Evt$ are called* dependent *if $(\alpha, \alpha') \notin I_{A,PV}$.*

Note that two visible events are called dependent regardless of whether they are weakly independent or not. This definition is motivated by the fact that $\mathbf{LTL}_{-\mathbf{X}}$ cannot distinguish between two sequences that differ in the ordering of two independent events (according to our definition).

We assume in the rest of the paper that a suitable subset PV is given to parameterize all the notions of equivalence, and omit the subscript PV whenever clear from the context.

3. Reasoning about Agents' Abilities

Many important properties in a MAS can be specified in terms of the strategic ability (or inability) of some agents to achieve a given goal. Such properties can be specified by formulas of the strategic logic **ATL**. The semantics of **ATL** is typically defined for models of synchronous systems. In this section, we show how to adapt it to asynchronous MAS.

3.1 Alternating-Time Temporal Logic: Syntax

Alternating-time temporal logic (Alur et al., 1997, 2002) generalizes the branching-time temporal logic **CTL** (Clarke & Emerson, 1981) by replacing the 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 the temporal property γ. The formulas make use of unary temporal operators: "X" ("next"), "G" ("always from now on"), "F" ("now or sometime in the future"), and binary ones: U ("strong until", γ_1 holds until γ_2 becomes true), and

R ("release", logical dual of U: γ_2 holds until and including the state when γ_1 becomes true; γ_1 is not required to ever hold, in which case γ_2 must be always true). The logic comes in several syntactic variants, the most popular of which are **ATL*** and **ATL**.

Definition 7 (Syntax of **ATL***)**.** *Let $\mathcal{PV}$ be a set of propositional variables and $\mathcal{A}$ the set of all agents. The language of* **ATL*** *is defined by the following grammar (where $\mathsf{p} \in \mathcal{PV}$ and $A \subseteq \mathcal{A}$):*

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

The other Boolean operators are defined as usual. "Release" 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 are given as $\mathrm{F}\,\gamma \equiv \mathit{true}\,\mathrm{U}\,\gamma$ and $\mathrm{G}\,\gamma \equiv \mathit{false}\,\mathrm{R}\,\gamma$. Moreover, the **CTL*** operator "for all paths" can be defined as $\mathsf{A}\gamma \equiv \langle\langle\emptyset\rangle\rangle\gamma$.

Definition 8 (Syntax of **ATL**)**.** *In* **ATL***, every occurrence of a strategic modality is immediately followed by a single temporal operator. In that case, "release" is not definable from "until" anymore (Laroussinie et al., 2008), and it must be added explicitly to the syntax as another primitive operator. Formally, the language of* **ATL** *is defined by the following grammar:*

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

In the rest of the paper, we are mainly interested in formulas that do not use the next step operator X, and do not contain nested strategic modalities. We denote the corresponding subsets of **ATL*** and **ATL** by **sATL*** ("simple **ATL***") and **sATL** ("simple **ATL**"). Moreover, **1ATL*** is the fragment of **sATL*** that admits only formulas consisting of a single strategic modality followed by an **LTL** formula (i.e., $\langle\langle A\rangle\rangle\gamma$, where $\gamma \in$ **LTL**), and analogously for **1ATL**.

Example 3. *The following formulas of* **sATL*** *specify interesting properties of the TGC system: $\langle\langle c\rangle\rangle\mathrm{F}\ \mathsf{in}_1$ (the controller can let train t_1 in), $\langle\langle c\rangle\rangle\mathrm{G}\,\neg\mathsf{in}_1$ (the controller can keep t_1 out forever), $\langle\langle c\rangle\rangle\mathrm{F}\,(\mathsf{in}_1 \wedge \mathrm{F}\,\neg\mathsf{in}_1)$ (the controller can let t_1 through), $\neg\langle\langle t_1, t_2\rangle\rangle\mathrm{F}\,(\mathsf{in}_1 \vee \mathsf{in}_2)$ (neither train can get in without the help of the controller, even if it collaborates with the other train).*

We claim that most of practically interesting specifications of strategic ability can be expressed in **sATL***, possibly extended with epistemic operators. Typically, one wants to verify if a group of agents can reach a "winning" state (expressed by formula $\langle\langle A\rangle\rangle\mathrm{F}\,\mathsf{win}$), or that they can avoid failure (captured by $\langle\langle A\rangle\rangle\mathrm{G}\,\neg\mathsf{fail}$). Nested strategic modalities allow to express that agents A can endow agents B with the ability to enforce some goal γ (captured by $\langle\langle A\rangle\rangle\mathrm{F}\,\langle\langle B\rangle\rangle\gamma$) or to deprive the other agents of the ability ($\langle\langle A\rangle\rangle\mathrm{G}\,\neg\langle\langle B\rangle\rangle\gamma$), which is seldom of practical interest.

3.2 Strategies and Outcomes

Let M be a model. A *strategy* of agent $i \in \mathcal{A}$ in M 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 (2004), 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*. Note that verification of **ATL** for agents with perfect recall is in general undecidable (Dima & Tiplea, 2011). Because of that, we focus on memoryless strategies, i.e., consider strategies of types Ir and ir. Formally:

- A *memoryless perfect information strategy for agent* i is a function $\sigma_i\colon St \to Evt_i$ such that $\sigma_i(g) \in P_i(g^i)$ for each global state $g \in St$.
- A *memoryless imperfect information strategy for* i is a function $\sigma_i\colon L_i \to Evt_i$ such that $\sigma_i(l) \in P_i(l)$ for each local state $l \in L_i$.

Thus, a perfect information strategy can assign different events to any two global states, while under imperfect information the agent's choices depend only on the local state of the agent.[2] A *joint strategy* σ_A for a coalition $A \subseteq \mathcal{A}$ is a tuple of strategies, one per agent $i \in A$. We denote the set of A's joint memoryless perfect (resp. imperfect) information strategies by Σ_A^{Ir} (resp. Σ_A^{ir}). Additionally, let $\sigma_A = (\sigma_1, \ldots, \sigma_k)$ be a joint strategy for $A = \{i_1, \ldots, i_k\}$. For each $g \in St$, we define $\sigma_A(g) = (\sigma_1(g), \ldots, \sigma_k(g))$.

Definition 9 (Outcome paths)**.** *Let* $Y \in \{\mathrm{Ir}, \mathrm{ir}\}$. *The* outcome *of strategy* $\sigma_A \in \Sigma_A^Y$ *in state* $g \in St$ *is the set* $out_M(g, \sigma_A) \subseteq \Pi_M(g)$ *such that* $\pi = g_0\alpha_0 g_1\alpha_1 g_2 \cdots \in out_M(g, \sigma_A)$ *iff* $g_0 = g$ *and* $\forall i \in \mathcal{N}\ \forall j \in A$ *if* $j \in Agent(\alpha_i)$, *then* $\alpha_i \in \sigma_j(\pi[i])$ *for* $Y = \mathrm{Ir}$, *and* $\alpha_i \in \sigma_j(\pi[i]^j)$ *for* $Y = \mathrm{ir}$.

Intuitively, the outcome of a joint strategy σ_A in a global state g is the set of all the infinite paths that can occur when in each state of the paths either some agents (an agent) in A execute(s) an event according to σ_A or some agents (an agent) in $\overline{A}$ execute(s) an event following their protocols. Clearly, each event α has to be executed by all agents which have α in their sets of events. In reasoning about asynchronous systems, one often wants to look only at *fair* paths, i.e., ones that do not consistently ignore an agent whose choice is always enabled. Formally, a path π satisfies the *concurrency-fairness condition* (**CF**) if there is no event enabled in all states of π from $\pi[i]$ on, and at the same time weakly independent from all the events actually executed in $\pi[i], \pi[i+1], \pi[i+2], \ldots$. We denote the set of all such paths starting at g by $\Pi_M^{CF}(g)$.

Definition 10 (**CF**-outcome)**.** *The* concurrency-fair outcome *of* $\sigma_A \in \Sigma_A^Y$ *is defined as* $out_M^{CF}(g, \sigma_A) = out_M(g, \sigma_A) \cap \Pi_M^{CF}(g)$.

3.3 Asynchronous Semantics of ATL and ATL*

Our semantics of **ATL*** for asynchronous interaction, parameterized with the strategy type $Y \in \{\mathrm{Ir}, \mathrm{ir}\}$, is defined as follows:

$M, g \models_Y \mathsf{p}$ iff $\mathsf{p} \in V(g)$, for $\mathsf{p} \in \mathcal{PV}$;

$M, g \models_Y \neg\varphi$ iff $M, g \not\models_Y \varphi$;

2. Alternatively, we can require the agent's choices to be the same for the global states that share the same local states.

$M, g \models_Y \varphi_1 \wedge \varphi_2$ iff $M, g \models_Y \varphi_1$ and $M, g \models_Y \varphi_2$;

$M, g \models_Y \langle\langle A \rangle\rangle \gamma$ iff there is a strategy $\sigma_A \in \Sigma_A^Y$ such that $out_M(g, \sigma_A) \neq \emptyset$ and, for each path $\pi \in out_M(g, \sigma_A)$, we have $M, \pi \models_Y \gamma$;

$M, \pi \models_Y \varphi$ iff $M, \pi[0] \models_Y \varphi$;

$M, \pi \models_Y \neg\gamma$ iff $M, \pi \not\models_Y \gamma$;

$M, \pi \models_Y \gamma_1 \wedge \gamma_2$ iff $M, \pi \models_Y \gamma_1$ and $M, \pi \models_Y \gamma_2$;

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

$M, \pi \models_Y \gamma_1 \,\mathrm{U}\, \gamma_2$ iff $M, \pi[i, \infty] \models_Y \gamma_2$ for some $i \geq 0$ and $M, \pi[j, \infty] \models_Y \gamma_1$ for all $0 \leq j < i$.

As usual, the semantics of **ATL** can be given entirely with respect to states:

Definition 11 (State-based semantics of **ATL**). *The Y-semantics of* **ATL** *can be equivalently defined by the following clauses:*

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

$M, g \models_Y \neg\varphi$ *iff* $M, g \not\models_Y \varphi$;

$M, g \models_Y \varphi_1 \wedge \varphi_2$ *iff* $M, g \models_Y \varphi_1$ *and* $M, g \models_Y \varphi_2$;

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

$M, g \models_Y \langle\langle A \rangle\rangle \varphi_1 \,\mathrm{U}\, \varphi_2$ *iff there is $\sigma_A \in \Sigma_A^Y$ st. $out_M(g, \sigma_A) \neq \emptyset$ and, for each $\pi \in out_M(g, \sigma_A)$, there exists $i \geq 0$ with $M, \pi[i] \models_Y \varphi_2$ and $M, \pi[j] \models_Y \varphi_1$ for all $0 \leq j < i$;*

$M, g \models_Y \langle\langle A \rangle\rangle \varphi_1 \,\mathrm{R}\, \varphi_2$ *iff there is $\sigma_A \in \Sigma_A^Y$ st. $out_M(g, \sigma_A) \neq \emptyset$ and, for all $\pi \in out_M(g, \sigma_A)$ and $i \geq 0$, either $M, \pi[i] \models_Y \varphi_2$ or $M, \pi[j] \models_Y \varphi_1$ for some $0 \leq j < i$.*

Example 4. *We leave it to the reader to check that all the formulas in Example 3 hold in the TGC model from Figure 2 (left) for both the* Ir *and the* ir *semantics.*

Remark 1. *We observe that the relation $\models_{\mathrm{ir}}$ captures the "objective" notion of ability under imperfect information (Jamroga, 2003; Bulling & Jamroga, 2014). That is, $\langle\langle A \rangle\rangle \gamma$ holds iff agents in A have a joint strategy to enforce γ from the current global state of the system. We expect to obtain analogous results for the semantics based on "subjective" ability (Schobbens, 2004; Jamroga & van der Hoek, 2004; Bulling & Jamroga, 2014), but a detailed study is outside the scope of this paper.*

Remark 2. *Note also that the semantics constrains the abilities behind $\langle\langle A \rangle\rangle$ to "no-deadlock" paths and strategies. That is, we only consider infinite execution paths, and only strategies whose outcomes are nonempty sets of such paths. This is in line with the standard approach to analysis of distributed systems. An interesting alternative would be to model executions with deadlock by paths ending with an infinite sequence of "silent" events, looping in the deadlock state. We plan to study the resulting semantics of* **ATL****, and model reductions for the semantics, in the future.*

We obtain the *concurrency-fair semantics* $\models_{\mathrm{IrF}}$ and $\models_{\mathrm{irF}}$ of **ATL** and **ATL*** by replacing $out_M(g, \sigma_A)$ with $out^{CF}_M(g, \sigma_A)$ in the clauses for $\langle\langle A \rangle\rangle$.

For the set of formulas $\mathcal{L}$ and the semantic relation $\models_Y$, we denote the logical system $(\mathcal{L}, \models_Y)$ by $\mathcal{L}_Y$. Thus, $\mathbf{ATL}_{\mathrm{Ir}}$ is **ATL** with memoryless perfect information semantics, $\mathbf{sATL}^*_{\mathrm{irF}}$ is the "simple **ATL***" with memoryless imperfect information strategies and concurrency-fairness assumption, and so on.

4. Model Checking sATL and sATL* for Asynchronous Agent Systems

In this work, we focus on simple specifications of strategic ability, i.e., ones that can be formally characterized without nesting strategic modalities. We believe that an overwhelming majority of properties, relevant in actual application domains, follow that pattern. One usually wants 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., see the example formulas below:

1. $\langle\langle i \rangle\rangle \mathrm{F}\, \mathsf{win_i}$,
2. $\langle\langle t_1, t_2 \rangle\rangle \mathrm{G}\, \neg(\mathsf{in_1} \wedge \mathsf{in_2})$,
3. $\langle\langle a, b \rangle\rangle \big(\mathrm{F}\, (\mathsf{knowsSecr_a} \wedge \mathsf{knowsSecr_b}) \wedge \mathrm{G}\, \neg \mathsf{knowsSecr_c}\big)$.

Note that (1) and (2) are formulas of **sATL** (in fact, **1ATL**), while (3) is a formula of **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. Moreover, in all realistic scenarios, players have only partial knowledge of the current global state of the world during the interaction. Thus, we focus here on the semantics based on imperfect information strategies.

In this section, we establish the complexity of model checking for some relevant fragments of $\mathbf{sATL}^*_{\mathrm{ir}}$ and $\mathbf{sATL}^*_{\mathrm{irF}}$. 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 compact *representation* of the system (in our case, an asynchronous MAS, cf. Section 2.1). We give both kinds of results. Note that $IIS_V(S)$ has usually exponentially many global states and transitions in the number of agents in S. Thus, the model checking results relative to the size of $IIS_V(S)$ "hide" the part of the complexity already included in the blowup. On the other hand, POR reduces models and not representations, so the complexity w.r.t. the size of the model tells us how much gain we can expect when the model is reduced.

We also briefly look at the *program complexity* of model checking, i.e., the complexity of the problem when the input formulas are of fixed or bounded length.

4.1 Model Checking $\mathbf{1ATL}_{\mathrm{ir}}$ and $\mathbf{1ATL}_{\mathrm{irF}}$

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

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

Proof. Analogous to the result by Schobbens (2004) for $\langle\langle\Gamma\rangle\rangle$-$\mathbf{ATL}_{\mathrm{ir}}$.

For the upper bound, we note that model checking of $\langle\langle A\rangle\rangle\gamma$ in M, g can be done by means of the following algorithm $\mathbf{mcheck_1}(\mathbf{M}, \mathbf{g}, \langle\langle\mathbf{A}\rangle\rangle\gamma)$:

1. Guess a joint ir-strategy σ_A;
2. Prune M according to σ_A, obtaining model M';
3. Model-check the **CTL** formula $\neg\mathsf{AG}\,\bot \wedge \mathsf{A}\gamma$ ("the set of paths is nonempty[3] and, for all paths, γ") in M', g, and return the answer.

Since σ_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. $|M|$, both with and without fairness assumptions (Baier & Katoen, 2008), we obtain the bound.

For the lower bound, we observe that single-agent systems can be seen as special cases of both synchronous and asynchronous systems, and the semantics with and without fairness assumptions coincide on such models. Moreover, there exists a polynomial reduction of the Boolean satisfiability problem (SAT) to model checking of the $\mathbf{ATL}_{\mathrm{ir}}$ formula $\langle\langle 1\rangle\rangle\mathrm{F}\,\mathsf{yes}$ in a single-agent model. Schobbens (2004, Section 3.1) describes the general idea of this reduction, and Jamroga (2015, Section 7.1.1) considers the single-agent version. This immediately gives us **NP**-hardness for model checking of $\mathbf{1ATL}_{\mathrm{ir}}$ and $\mathbf{1ATL}_{\mathrm{irF}}$. Note also that the lower bound does not rely on the length of the formula, as formulas of length 3 are sufficient to construct the reduction. □

Proposition 1 established the complexity of model checking when the input model is given explicitly. We now examine the complexity of the same problem with respect to the size of representations, i.e., AMAS. Since the proofs are more involved than for Proposition 1, we will separately prove the upper and the lower bounds.

Proposition 2. *Model checking* $\mathbf{1ATL}_{\mathrm{ir}}$ *is in* **PSPACE** *with respect to the size of the representation.*

Proof. Observe that model checking of formula $\langle\langle A\rangle\rangle\gamma$ in an asynchronous MAS S can be done by means of algorithm $\mathbf{mcheck_2}(\mathbf{S}, \langle\langle\mathbf{A}\rangle\rangle\gamma)$:

1. Guess a joint ir-strategy σ_A as a deterministic restriction of the protocols $P_i, i \in A$. That is, for every agent i in the coalition, and every local state $l_i \in L_i$, select nondeterministically an event $\alpha_{(i,l_i)} \in P_i(l_i)$;
2. Prune S according to σ_A, obtaining AMAS S'. That is, update P'_i for all $i \in A$, so that $P'_i(l_i) = \{\alpha_{(i,l_i)}\}$;
3. Model-check the **LTL** formula $\mathrm{G}\,\bot$ in the resulting representation S' (it holds only if the set of paths is empty), and return **false** if the formula is true;
4. Otherwise, model-check the **LTL** formula γ in S', and return the answer.

Since the size of σ_A is linear w.r.t. $|S|$, and model checking **LTL** is in **PSPACE** w.r.t. $|S|$ (Schnoebelen, 2003), we obtain the bound. □

3. Note that $\mathsf{AG}\,\bot$ can hold only if the set of outgoing paths is empty.

Proposition 3. *Model checking* $\mathbf{1ATL}_{\mathrm{irF}}$ *is in* **PSPACE** *with respect to the size of the representation.*

Proof. Verification in concurrency-fair paths is more sophisticated, and requires an additional construction. Given an asynchronous MAS S with agents $A_i^S = (L_i^S, \iota_i^S, Evt_i^S, P_i^S, T_i^S)$, we construct its *event trail extension* $ETrail(S)$ by adding, for each A_i^S, a new state (l, α) for every local state $l \in L_i^S$ and every event $\alpha \in Evt_i^S$ that labels some incoming transition in l. We also modify the transition function T_i^S so that the outcome of each transition "records" the latest event, i.e., $T_i^{ETrail(S)}(l, \alpha) = (T_i^S(l), \alpha)$ and $T_i^{ETrail(S)}((l, \alpha), \alpha') = (T_i^S(l), \alpha')$. The valuation of the atomic propositions in $\mathcal{PV}$ carries over to the event trail extension, except for new atomic propositions evt_α, one per $\alpha \in Evt_i$, that label states where event α has just been executed. That is, $V_i^{ETrail(S)}(l) = V_i^S(l)$, and $V_i^{ETrail(S)}((l, \alpha)) = V_i^S(l) \cup \{\mathsf{evt}_\alpha\}$. Finally, for each pair of events $\alpha, \beta \in Evt^S$, we add new proposition $\mathsf{independent}_{\alpha,\beta}$ which holds in all the local states of S if α, β are weakly independent in S, and otherwise does not hold anywhere.

Note that the $\mathbf{CTL}^*$ formula $enabled_\alpha \equiv \mathsf{EX}\ \mathsf{evt}_\alpha$ expresses that event α is enabled in the current global state of the system. Moreover, the $\mathbf{CTL}^*$ path formula $alwaysEnabled_\alpha \equiv \mathrm{G}\ enabled_\alpha$ says that α is always enabled on the selected path. Likewise, $indExec_\alpha \equiv \mathrm{X}\,\mathrm{G}\,\big(\bigvee_{\beta \in Evt}(\mathsf{evt}_\beta \wedge \mathsf{independent}_{\alpha,\beta})\big)$ says that only events independent from α will be executed. Consequently, $fair \equiv \bigwedge_\alpha \mathrm{G}\,(alwaysEnabled_\alpha \rightarrow \neg indExec_\alpha)$ expresses that the selected path satisfies concurrency-fairness.

Now, model checking of formula $\langle\!\langle A \rangle\!\rangle \gamma$ in an asynchronous MAS S can be done by algorithm $\mathbf{mcheck_3}(\mathbf{S}, \langle\!\langle \mathbf{A} \rangle\!\rangle \gamma)$:

1. Guess a joint ir-strategy σ_A as a deterministic restriction of the protocols $P_i, i \in A$;
2. Prune S according to σ_A, obtaining AMAS S';
3. Model-check the $\mathbf{CTL}^*$ formula $\big(\mathsf{E}\, fair \ \wedge\ \mathsf{A}\,(fair \rightarrow \gamma)\big)$ in the event trail extension of the resulting representation, i.e., in $ETrail(S')$, and return the answer.

Since the size of σ_A is linear w.r.t. $|S|$, and model checking $\mathbf{CTL}^*$ is in **PSPACE** w.r.t. $|S|$ (Schnoebelen, 2003), we obtain the bound. □

Proposition 4. *Model checking* $\mathbf{1ATL}_{\mathrm{ir}}$ *and* $\mathbf{1ATL}_{\mathrm{irF}}$ *is* **PSPACE***-hard in the size of the representation (even for formulas of bounded length).*

Proof. To prove the lower bound, we adapt the construction by Kupferman et al. (2000, Theorem 6.1). Given a Turing machine T with space complexity $s(n)$, a concurrent program $P(T)$ of size $O(s(n))$ is constructed, such that accepting termination of T is reduced to model checking of the **CTL** formula $\mathsf{EF}\ \mathsf{accept}$ in $P(T)$. We recall from Kupferman et al. (2000) that a concurrent program can be seen as a collection of local automata with disjoint event sets, executed synchronously. That is, all the automata make a synchronous step at every tick of the global clock. According to Kupferman et al. (2000, Theorem 6.1), there exists a computation of T on the empty tape which eventually reaches an accepting state iff $P(T) \models_{\mathbf{CTL}} \mathsf{EF}\ \mathsf{accept}$.

We also observe that, for every synchronous program P, one can obtain an asynchronous MAS $Async(P)$ with a similar behaviour by sequentializing the concurrent events of modules

in P in an arbitrary order, and adding an extra "synchronizer" agent which enforces that each module $i \in \{1, \ldots, n\}$ takes the ith step in every "execution cycle." In consequence, each concurrent transition in P is decomposed into a sequence of n asynchronous transitions in $Async(P)$. Clearly, $\mathsf{EF}\ \mathsf{p}$ holds in P iff it holds in $Async(P)$. Thus, we have that there exists an accepting computation of T on the empty tape iff $P(T) \models_{\mathbf{CTL}} \mathsf{EF}\ \mathsf{accept}$ iff $IIS(Async(P(T))) \not\models_{\mathrm{ir}} \langle\langle\emptyset\rangle\rangle \mathrm{G}\, \neg\mathsf{accept}$. This way we obtain the **co-PSPACE**-hardness for $\mathbf{1ATL}_{\mathrm{ir}}$. Since **co-PSPACE** = **PSPACE**, the lower bound follows.

For $\mathbf{1ATL}_{\mathrm{irF}}$, we observe that all the paths in $IIS(Async(P(T)))$ are fair, so the same construction can be used.

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

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

4.2 Model Checking $\mathbf{sATL}_{\mathrm{ir}}$ and $\mathbf{sATL}_{\mathrm{irF}}$

The verification complexity for Boolean combinations of formulas from **1ATL** is almost the same.

Proposition 5. *Model checking* $\mathbf{sATL}_{\mathrm{ir}}$ *and* $\mathbf{sATL}_{\mathrm{irF}}$ *is* **NP***-hard and in* $\mathbf{\Theta_2^P}$ *in the size of the model and the length of the formula (even for formulas of bounded length).*[4]

Proof. The lower bound follows from Proposition 1. The upper bound is demonstrated by the algorithm $\mathbf{mcheck_5}(\mathbf{M}, \mathbf{g}, \varphi)$ below:

1. If $\varphi \equiv \langle\langle A\rangle\rangle\gamma$, then return $\mathbf{mcheck_1}(\mathbf{M}, \mathbf{g}, \langle\langle \mathbf{A}\rangle\rangle\gamma)$;
2. If $\varphi \equiv \neg\psi$, then return (not $\mathbf{mcheck_5}(\mathbf{M}, \mathbf{g}, \psi)$);
3. If $\varphi \equiv \psi_1 \wedge \psi_2$, then return ($\mathbf{mcheck_5}(\mathbf{M}, \mathbf{g}, \psi_\mathbf{1})$ and $\mathbf{mcheck_5}(\mathbf{M}, \mathbf{g}, \psi_\mathbf{2})$);

Thus, the non-deterministic algorithm in Proposition 1 is used as an oracle to establish the truth value for each subformula $\langle\langle A\rangle\rangle\gamma$ of φ. Clearly, 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 in the standard way.

We suspect that the problem is $\mathbf{\Theta_2^P}$-complete. We leave the proof for future work. □

Proposition 6. *Model checking* $\mathbf{sATL}_{\mathrm{ir}}$ *and* $\mathbf{sATL}_{\mathrm{irF}}$ *is* **PSPACE***-complete in the size of the representation and the length of the formula (even for formulas of bounded length).*

Proof. Analogous to Proposition 5. The lower bound follows from Proposition 4. For the upper bound, we use algorithm $\mathbf{mcheck_6}(\mathbf{M}, \mathbf{g}, \varphi)$:

1. If $\varphi \equiv \langle\langle A\rangle\rangle\gamma$, then return $\mathbf{mcheck_2}(\mathbf{M}, \mathbf{g}, \langle\langle \mathbf{A}\rangle\rangle\gamma)$ when model checking $\mathbf{sATL}_{\mathrm{ir}}$ and $\mathbf{mcheck_3}(\mathbf{M}, \mathbf{g}, \langle\langle \mathbf{A}\rangle\rangle\gamma)$ for $\mathbf{sATL}_{\mathrm{irF}}$;
2. If $\varphi \equiv \neg\psi$, then return (not $\mathbf{mcheck_6}(\mathbf{M}, \mathbf{g}, \psi)$);
3. If $\varphi \equiv \psi_1 \wedge \psi_2$, then return ($\mathbf{mcheck_6}(\mathbf{M}, \mathbf{g}, \psi_\mathbf{1})$ and $\mathbf{mcheck_6}(\mathbf{M}, \mathbf{g}, \psi_\mathbf{2})$);

Thus, we now use the algorithms from Propositions 2 and 3 as the oracles. Since $\mathbf{P^{PSPACE}} =$ **PSPACE**, we obtain the result. □

4. Where $\mathbf{\Theta_2^P} = \mathbf{P^{||NP}}$ is the class of problems solvable by a deterministic polynomial-time Turing machine making polynomially many *nonadaptive* calls to an **NP** oracle.

4.3 Model Checking $\mathbf{sATL}^*_{\mathrm{ir}}$ and $\mathbf{sATL}^*_{\mathrm{irF}}$

Finally, we examine the complexity of verification for specifications with arbitrary **LTL** subformulas.

Proposition 7. *Model checking* $\mathbf{1ATL}^*_{\mathrm{ir}}$, $\mathbf{1ATL}^*_{\mathrm{irF}}$, $\mathbf{sATL}^*_{\mathrm{ir}}$, *and* $\mathbf{sATL}^*_{\mathrm{irF}}$ *is* **PSPACE***-complete in the size of the model and the length of the formula.*

Proof. The lower bound follows from **PSPACE**-completeness of **LTL** model checking (Schnoebelen, 2003). The upper bound for $\mathbf{1ATL}^*_{\mathrm{ir}}$ can be obtained by the following algorithm $\mathbf{mcheck_7}(\mathbf{M}, \mathbf{g}, \langle\langle \mathbf{A} \rangle\rangle \gamma)$:

1. Guess a joint ir-strategy σ_A;
2. Prune M according to σ_A, obtaining model M';
3. Model-check the **LTL** formula $\mathrm{G} \bot$ in M', g (it holds only if the set of paths is empty), and return **false** if the formula is true;
4. Otherwise, model-check the **LTL** formula γ in M', g, and return the answer.

For $\mathbf{1ATL}^*_{\mathrm{irF}}$, we use an analogous construction to the one for Proposition 3. Given an IIS M, we construct its *event trail extension* $ETrail(M)$ analogously, i.e., for every state q in M and every incoming transition in q, labeled by α, we add a new state (q, α) being essentially a copy of q, and labeled additionally by a fresh proposition $\mathtt{evt}_\alpha$. Formula *fair* is defined as in Proposition 3. Now, model checking of $\langle\langle A \rangle\rangle \gamma$ in M, q can be done by algorithm $\mathbf{mcheck_{7F}}(\mathbf{M}, \mathbf{q}, \langle\langle \mathbf{A} \rangle\rangle \gamma)$:

1. Guess a joint ir-strategy σ_A;
2. Prune M according to σ_A, obtaining model M';
3. Model-check the **CTL*** formula $\big(\mathsf{E}\, \mathit{fair} \wedge \mathsf{A}\,(\mathit{fair} \rightarrow \gamma)\big)$ in $ETrail(M'), q$, and return the answer.

Finally, for $\mathbf{sATL}^*_{\mathrm{ir}}$ and $\mathbf{sATL}^*_{\mathrm{irF}}$, we repeat the above procedure for each subformula, and compute the Boolean combination, i.e., use an analogous algorithm to that in Proposition 5. Since $\mathbf{P}^{\mathbf{PSPACE}} = \mathbf{PSPACE}$, we obtain the upper bound. □

Proposition 8. *For formulas of bounded length, model checking is* **NP***-complete for* $\mathbf{1ATL}^*_{\mathrm{ir}}$ *and* $\mathbf{1ATL}^*_{\mathrm{irF}}$*, and between* **NP** *and* $\mathbf{\Theta_2^P}$ *for* $\mathbf{sATL}^*_{\mathrm{ir}}$ *and* $\mathbf{sATL}^*_{\mathrm{irF}}$ *in the size of the model.*

Proof. The lower bounds follow from Proposition 1.

Regarding the upper bounds, recall that **CTL*** model checking is in **P** for formulas of bounded length (Schnoebelen, 2003). Thus, we have that the algorithm $\mathbf{mcheck_7}(\mathbf{M}, \mathbf{g}, \langle\langle \mathbf{A} \rangle\rangle \gamma)$, that implements model checking of $\mathbf{sATL}^*_{\mathrm{ir}}$, runs in deterministic polynomial time with non-adaptive queries to an **NP** oracle. When only one non-negated strategic modality is allowed (i.e., for $\mathbf{1ATL}^*_{\mathrm{ir}}$), the complexity of the algorithm is simply **NP**.

The same applies to the algorithm $\mathbf{mcheck_{7F}}(\mathbf{M}, \mathbf{g}, \langle\langle \mathbf{A} \rangle\rangle \gamma)$ for model checking $\mathbf{sATL}^*_{\mathrm{irF}}$ and $\mathbf{1ATL}^*_{\mathrm{irF}}$. □

Proposition 9. *Model checking* $\mathbf{sATL}^*_{\mathrm{ir}}$, $\mathbf{1ATL}^*_{\mathrm{ir}}$, $\mathbf{sATL}^*_{\mathrm{irF}}$, *and* $\mathbf{1ATL}^*_{\mathrm{irF}}$ *is* **PSPACE***-complete in the size of the representation and the length of the formula (even for the formulas of bounded length).*

Proof. The lower bounds follow from Proposition 4. The upper bounds are obtained analogously to Proposition 6, with algorithm $\mathbf{mcheck_7}$ serving as the oracle. □

4.4 Discussion

The above complexity results show that model checking fragments of $\mathbf{sATL}^*_{\mathrm{ir}}$ and $\mathbf{sATL}^*_{\mathrm{irF}}$ with respect to asynchronous multi-agent systems is hard, and the size of the representation is the main factor for this hardness. Moreover, it is generally believed that fixpoint equivalences do not hold for strategic abilities with imperfect information (Bulling & Jamroga, 2011; Dima et al., 2014). Thus, to model check AMAS we must, in many cases, resort to unfolding the representation into an explicit model (i.e., an interleaved interpreted system), and then verifying the IIS. Because of that, it is essential for the unfolding to *produce as small models 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 what follows, we recall the idea of *partial order reduction*, very important in verification of temporal properties in asynchronous systems, and show how it can be used to model-check formulas of $\mathbf{sATL}^*_{\mathrm{ir}}$ and $\mathbf{sATL}^*_{\mathrm{irF}}$.

5. Partial Order Reductions

Partial order reductions (POR) have been defined for various configurations of temporal and temporal-epistemic logics without the "next step" operator X (Peled, 1993; Penczek et al., 2000; Gerth et al., 1999; Lomuscio et al., 2010a, 2010b). The idea is to generate reduced models that either preserve some kind of model equivalence, or preserve representatives of Mazurkiewicz traces (Mazurkiewicz, 1977). The former method was used, for instance, to construct POR for $\mathbf{LTL}_{-\mathbf{X}}$ and $\mathbf{LTLK}_{-\mathbf{X}}$ based on stuttering trace equivalence (Lomuscio et al., 2010a, 2010b), and to obtain reductions for $\mathbf{CTL}^*_{-\mathbf{X}}$ and $\mathbf{CTLK}_{-\mathbf{X}}$ based on stuttering bisimulation (Gerth et al., 1999; Lomuscio et al., 2010a, 2010b). The latter method was applied e.g. to prove correctness of reduction for $\mathbf{LTL}_{-\mathbf{X}}$ formulas under the concurrency-fair semantics (Peled, 1993).

It is essential to note that the practical value of a reduction scheme depends on how discriminative the underlying notion of equivalence between paths is. Since $\mathbf{CTL}_{-\mathbf{X}}$ equivalences are more discriminative than those for $\mathbf{LTL}_{-\mathbf{X}}$, there are more equivalence classes for $\mathbf{CTL}_{-\mathbf{X}}$, and more paths need to be retained in the reduced model as representatives. In consequence, partial order reductions preserving $\mathbf{LTL}_{-\mathbf{X}}$ produce smaller models than those for $\mathbf{CTL}_{-\mathbf{X}}$. Note that $\mathbf{ATL}^*_{-\mathbf{X}}$ and $\mathbf{ATL}_{-\mathbf{X}}$ (i.e., $\mathbf{ATL}^*$ and $\mathbf{ATL}$ without the next step operator X) have even more distinguishing power than $\mathbf{CTL}_{-\mathbf{X}}$. Thus, one can expect that equivalences preserving full $\mathbf{ATL}^*_{-\mathbf{X}}$ would be very discriminative, and result in very inefficient reductions. Aware of this and motivated by practical applications, we do not look for a general POR for $\mathbf{ATL}^*_{-\mathbf{X}}$ in this paper. Instead, we look for *subsets of* $\mathbf{ATL}^*_{-\mathbf{X}}$ *for which the most efficient known partial order reduction methods* (i.e., those for $\mathbf{LTL}_{-\mathbf{X}}$) *can be applied.*

Note that $\mathbf{1ATL}^*$ is an extension of $\mathbf{LTL}_{-\mathbf{X}}$ with a single strategic modality, which is put at the beginning of an $\mathbf{LTL}_{-\mathbf{X}}$ formula. $\mathbf{sATL}^*$ extends this further to Boolean combinations of such simple formulas. Therefore, $\mathbf{sATL}^*$ has only slightly more distinguishing

power that $\mathbf{LTL_{-X}}$, which is one of the reasons that POR of $\mathbf{LTL_{-X}}$ work relatively directly for $\mathbf{sATL^*_{ir}}$. Our conjecture is that POR of $\mathbf{LTL_{-X}}$ preserve a slightly stronger equivalence that the one induced by $\mathbf{LTL_{-X}}$. On the other hand, the discriminative powers of $\mathbf{sATL^*}$ and $\mathbf{CTL_{-X}}$ are not comparable.

In what follows, we show that the reductions for $\mathbf{LTL_{-X}}$ can be adapted to $\mathbf{sATL^*_{ir}}$ and its fragments, both with and without the **CF** assumption. We begin by introducing the relevant notions of equivalence (Sections 5.2 and 5.3). Then, we propose conditions on reduced models that preserve the equivalences (Sections 5.4 and 5.5). Finally, we present algorithms for POR and show their correctness (Section 6).

Interestingly, it turns out that our approach does not apply to $\mathbf{sATL^*_{Ir}}$, cf. Section 6.3. This suggests that **ATL** with imperfect information, besides conceptual advantage, can possibly offer some technical benefits over **ATL** with perfect information.

5.1 Properties of Submodels

We first state the following two lemmas regarding the relation between the outcome set of a strategy in a full model and that in its submodel.[5]

Lemma 1. *Let M' be a submodel of M. For each* ir*-joint strategy σ_A we have $out_{M'}(\iota, \sigma_A) = out_M(\iota, \sigma_A) \cap \Pi_{M'}(\iota)$ and $out^{CF}_{M'}(\iota, \sigma_A) = out^{CF}_M(\iota, \sigma_A) \cap \Pi^{CF}_{M'}(\iota)$.*

Proof. Note that each ir-strategy in M is also a well defined ir-strategy in M' as it is defined on the local states of AMAS which is extended by M and M'. The lemma follows directly from Definitions 9 and 10, together with the fact that $\Pi_{M'}(\iota) \subseteq \Pi_M(\iota)$. □

The analogous lemma holds also for Ir-joint strategies, but σ_A of M needs to be restricted to the global states of M'. The next lemma says that paths which follow the same sequence of events from agent i's perspective cannot be distinguished by any strategy of i.

Lemma 2. *Let M be a model, $\pi, \pi' \in \Pi_M(\iota)$, and for some $i \in \mathcal{A}$: $Evt(\pi) \mid_{Evt_i} = Evt(\pi') \mid_{Evt_i}$. Then, for each* ir*-strategy σ_i, we have $\pi \in out_M(\iota, \sigma_i)$ iff $\pi' \in out_M(\iota, \sigma_i)$.*

Proof. Let $\pi = g_0\alpha_0 g_1\alpha_1 g_2\alpha_2 \ldots$ and $Evt(\pi) \mid_{Evt_i} = \alpha_{i_0}\alpha_{i_1}\alpha_{i_2}\ldots$ be the sequence of events of agent i in π. This sequence can be either finite or infinite. If it is empty, then the thesis trivially holds. So, assume that $Evt(\pi) \mid_{Evt_i}$ is not empty. Let L be equal to the length of $Evt(\pi) \mid_{Evt_i}$ if $Evt(\pi) \mid_{Evt_i}$ is finite or be equal to ∞ otherwise. For each α_{i_j} let $\pi[\alpha_{i_j}] = \pi[i_j] = g_{i_j}$ denote the global state from which α_{i_j} is executed in π, where $0 \leq j < L$.

By induction we can show that for each $0 \leq j < L$ we have $\pi[\alpha_{i_j}]^i = \pi'[\alpha_{i_j}]^i$. For $j = 0$ it is easy to note that $\pi[\alpha_{i_0}]^i = \pi'[\alpha_{i_0}]^i = \iota^i$, which follows from the fact that the paths π and π' start at the same global state ι and $Evt(\pi) \mid_{Evt_i} = Evt(\pi') \mid_{Evt_i}$.

Assume that the thesis holds for $j = k$. The induction step follows from the fact the local evolution T_i is a function, so if $\pi[\alpha_{i_k}]^i = \pi'[\alpha_{i_k}]^i = l$ for some $l \in L_i$, then $\pi[\alpha_{i_{k+1}}]^i = \pi'[\alpha_{i_{k+1}}]^i = T_i(l, \alpha_{i_k})$. So, the events of Evt_i are executed from the same local states in π and π', which means that $\alpha_{i_j} \in \sigma_i(\pi[i_j]^i)$ iff $\alpha_{i_j} \in \sigma_i(\pi'[i_j]^i)$ for $0 \leq j < L$.

5. In line with the terminology established in the previous part of the paper, we use the term "model" to refer to an interleaved interpreted system, see Section 2.2 for the precise definitions.

Consequently, for each ir-strategy σ_i, we have $\pi \in out_M(\iota, \sigma_i)$ iff $\pi' \in out_M(\iota, \sigma_i)$, which concludes the proof. □

The lemma can be easily generalized to joint strategies $\sigma_A \in \Sigma_A^{\mathrm{ir}}$. Note that the same property does not hold for perfect information strategies. This is because the current local state l_i can only change through the execution of an event by agent i, but the current global state can possibly change because of another agent's transition. Similarly, the analogue of Lemma 2 does not hold in synchronous models of MAS, since the local transitions of i in a synchronous model can be influenced by the events selected by the other agents.

5.2 Stuttering Equivalences for LTL$_{-\mathbf{X}}$

We now recall the definitions of stuttering equivalence and stuttering path equivalence. Let M be a model, $M' \subseteq M$, and $PV \subseteq \mathcal{PV}$ a subset of propositions. Stuttering equivalence says that two paths can be divided into corresponding finite segments, each satisfying exactly the same propositions. Stuttering path equivalence[6] requires two models to always have corresponding, stuttering-equivalent paths. Theorem 1 connects the latter to **LTL$_{-\mathbf{X}}$**.

Definition 12 (Stuttering equivalence (Clarke et al., 1999))**.** *Two paths $\pi \in \Pi_M(\iota)$ and $\pi' \in \Pi_{M'}(\iota)$ are* stuttering equivalent*, denoted $\pi \equiv_s \pi'$, if there exists a partition $B_0 = (\pi[0], \ldots, \pi[i_1 - 1])$, $B_1 = (\pi[i_1], \ldots, \pi[i_2 - 1])$, $\ldots$ of the states of π, and an analogous partition $B'_0, B'_1, \ldots$ of the states of π', such that for each $j \geq 0 : B_j$ and B'_j are nonempty and finite, and $V(g) \cap PV = V'(g') \cap PV$ for every $g \in B_j$ and $g' \in B'_j$.*

Definition 13 (Stuttering path equivalence (Clarke et al., 1999))**.** *Models M and M' are* stuttering path equivalent*, denoted $M \equiv_s M'$ if for each path $\pi \in \Pi_M(\iota)$, there is a path $\pi' \in \Pi_{M'}(\iota)$ such that $\pi \equiv_s \pi'$.*[7]

Theorem 1 (Clarke et al. (1999))**.** *If $M \equiv_s M'$, then, for any* **LTL$_{-\mathbf{X}}$** *formula φ over PV, we have $M, \iota \models \varphi$ iff $M', \iota' \models \varphi$.*

5.3 Independence-Based Equivalences

Partial order reductions for concurrency-fair **LTL$_{-\mathbf{X}}$** are based on *Mazurkiewicz traces* as introduced by Mazurkiewicz (1977) and used e.g. in his later works (Mazurkiewicz, 1986, 1988). By Evt^* (resp. Evt^ω), we denote the set of finite (resp. infinite) sequences of events. Consider two sequences $w, w' \in Evt^*$. We say that $w \sim_I w'$ iff $w = w_1 \alpha \alpha' w_2$ and $w' = w_1 \alpha' \alpha w_2$, for some $w_1, w_2 \in Evt^*$ and $(\alpha, \alpha') \in I_\emptyset$. Let $\equiv_I$ be the reflexive and transitive closure of $\sim_I$. By (finite) traces we mean the equivalence classes of $\equiv_I$, denoted by $[w]_{\equiv_I}$. Formally, the definition of a trace is $[w]_{\equiv_I} = \{w' \in Evt^* \mid w' \equiv_I w\}$.

To define infinite traces we need additional concepts. Let $v, v' \in Evt^\omega$, and let $Pref(v)$ denote the set of the finite prefixes of v. Now, the relation $\leq_I$ is defined as follows: $v \leq_I v'$ iff $\forall u \in Pref(v) \exists \hat{u} \in Pref(v) \exists u' \in Pref(v')(u \in Pref(\hat{u}) \wedge \hat{u} \equiv_I u')$. That is, each finite prefix of v can be extended to a permutation (under commuting adjacent independent events) of

6. The property is usually called *stuttering trace equivalence*. We opt for a slightly different name to avoid confusion with Mazurkiewicz traces, also used in this paper.
7. Typically, the definition contains also the symmetric condition which in our case always holds for M and its submodel M', as $\Pi_{M'}(\iota) \subseteq \Pi_M(\iota)$.

some prefix of v'. Moreover, let $v \equiv_I^\omega v'$ iff $v \leq_I v'$ and $v' \leq_I v$. Infinite traces are defined as equivalence classes of the relation $\equiv_I^\omega$, denoted by $[v]_{\equiv_I^\omega}$.

Theorem 2 (Peled (1996a))**.** *Let M be a model.*[8] *If $\pi, \pi' \in \Pi_M(\iota)$ such that $Evt(\pi) \equiv_I^\omega Evt(\pi')$, then $\pi \equiv_s \pi'$.*

Thus, paths over representatives of the same infinite trace cannot be distinguished by any $\mathbf{LTL_{-X}}$ formula using propositions of PV. Note that Mazurkiewicz traces preserve **CF**, i.e., if $\pi \in \Pi_M^{CF}(\iota)$, then for each π' such that $Evt(\pi) \equiv_I^\omega Evt(\pi')$ we have $\pi' \in \Pi_M^{CF}(\iota)$. This fact has been used to define partial order reductions for $\mathbf{LTL_{-X}}$ under the **CF** assumption. We will now generalize the approach to $\mathbf{sATL^*_{ir}}$.

5.4 Preserving Traces for $\mathbf{sATL^*_{irF}}$

Rather than generating the full model $M = IIS(S)$, one can generate a reduced model M' of M satisfying the following property:

$$\textbf{AE-CF}\colon\ (\forall \pi \in \Pi_M^{CF}(\iota))(\exists \pi' \in \Pi_{M'}^{CF}(\iota)) Evt(\pi) \equiv_I^\omega Evt(\pi').$$

Then, M' preserves the $\mathbf{LTL_{-X}}$ formulas under **CF** over PV (Peled, 1996a). We will now prove that this also works for $\mathbf{sATL^*_{irF}}$.

We first show that each set $out_M(g, \sigma_A)$ is trace-complete in the sense that with each path π such that $Evt(\pi) = w$, it contains a path over any $w' \in [w]_{\equiv_I^\omega}$.

Lemma 3. *Let $\pi \in out_M(\iota, \sigma_A)$ and $Evt(\pi) = w$. Then, $\forall w' \in [w]_{\equiv_I^\omega} \exists \pi' \in out_M(\iota, \sigma_A)$ such that $Evt(\pi') = w'$.*

Proof. Let M' be obtained from M by fixing $P_i(l_i) = \{\sigma_i(l_i)\}$ for each $i \in A, l_i \in L_i$, and pruning the transitions accordingly. That is, transitions of agents outside coalition A remain unchanged, while agents in A only keep those consistent with strategy σ_A. Consider the set of paths $\Pi_{M'}(\iota)$. Let w be a sequence of events obtained by traversing M' along some path π, i.e., $w = Evt(\pi)$. Following the inductive reasoning of Peled (1996a, Theorem 3.3), while reading w, an arbitrary equivalent sequence $w' \in \Pi_{M'}(\iota)$ can be produced. Thus, $\Pi_{M'}(\iota)$ is trace-complete. But, from the construction of M' and in accordance with Definition 9, we have that $\Pi_{M'}(\iota) = out_M(\iota, \sigma_A)$, which ends the proof. □

The above lemma implies the following.

Lemma 4. *Let M be a model and M' its submodel satisfying the property **AE-CF**. Then, for each* ir*-strategy σ_A, $\forall \pi \in out_M^{CF}(\iota, \sigma_A)\ \exists \pi' \in out_{M'}^{CF}(\iota, \sigma_A)$ such that $Evt(\pi) \equiv_I^\omega Evt(\pi')$.*

Proof. Assume that $\pi \in out_M^{CF}(\iota, \sigma_A)$. Then there is $\pi' \in \Pi_{M'}^{CF}(\iota)$ such that $Evt(\pi) \equiv_I^\omega Evt(\pi')$ (by **AE-CF**). Since M' is a submodel of M, we have that $\pi' \in \Pi_M^{CF}(\iota)$. This implies that $\pi' \in out_M^{CF}(\iota, \sigma_A)$ by Lemma 3. Since $\pi' \in \Pi_{M'}^{CF}(\iota)$ by Definition 9, we obtain that $\pi' \in out_{M'}^{CF}(\iota, \sigma_A)$, which together with the fact that $Evt(\pi) \equiv_I^\omega Evt(\pi')$ completes the proof. □

8. Technically, Peled's original theorem refers to traces in a *finite state program*. It remains applicable to IIS as defined in Sec. 2.2, since we consider sequences of events indiscriminately of agents and strategies.

Theorem 3. *Let M be a model and M' its submodel satisfying **AE-CF**. For each* $\mathbf{sATL}^*_{\mathrm{irF}}$ *formula φ over PV we have:*

$$M, \iota \models_{\mathrm{irF}} \varphi \quad \textit{iff} \quad M', \iota' \models_{\mathrm{irF}} \varphi.$$

Proof. Proof by induction on the structure of φ. We show the case $\varphi = \langle\langle A \rangle\rangle \gamma$. The cases for $\neg, \wedge$ are straightforward.

(⇒) Follows from the fact that for each ir-joint strategy σ_A we have $out^{CF}_{M'}(\iota, \sigma_A) = out^{CF}_{M}(\iota, \sigma_A) \cap \Pi^{CF}_{M'}(\iota)$, so $out^{CF}_{M'}(\iota, \sigma_A) \subseteq out^{CF}_{M}(\iota, \sigma_A)$.

(⇐) Assume that $M', \iota \models_{\mathrm{irF}} \langle\langle A \rangle\rangle \gamma$. From the semantics, there is an ir-joint strategy σ_A such that for each $\pi \in out^{CF}_{M'}(\iota, \sigma_A)$ we have $M', \pi \models_{\mathrm{irF}} \gamma$. In order to prove the thesis, we show that for each $\pi \in out^{CF}_{M}(\iota, \sigma_A) \setminus out^{CF}_{M'}(\iota, \sigma_A)$ we have $M, \pi \models_{\mathrm{irF}} \gamma$. It follows from Lemma 4 and Theorem 2 that for each $\pi \in out^{CF}_{M}(\iota, \sigma_A)$ there is $\pi' \in out^{CF}_{M'}(\iota, \sigma_A)$ such that $\pi \equiv_s \pi'$. So, $M', \pi' \models_{\mathrm{irF}} \gamma$ implies that $M, \pi \models_{\mathrm{irF}} \gamma$. Thus, we can conclude that $M, \iota \models_{\mathrm{irF}} \langle\langle A \rangle\rangle \gamma$. □

5.5 Stuttering Equivalence without CF

The method based on Mazurkiewicz traces works well for $\mathbf{sATL}^*_{\mathrm{irF}}$, and we will present an algorithm generating reduced models that satisfy condition **AE-CF** in Section 6. The same cannot be easily applied to the semantics without fairness. In particular, it is unclear how to generate reduced models that satisfy the analogue of **AE-CF** in all paths (and not only the fair ones). However, a similar result can be obtained for a relevant subset of $\mathbf{sATL}^*_{\mathrm{ir}}$ through stuttering equivalence, based on the following structural property:

$$\mathbf{AE}_A\colon \forall \sigma_A \in \Sigma^{\mathrm{ir}}_A \forall \pi \in out_M(\iota, \sigma_A)\ \exists \pi' \in out_{M'}(\iota, \sigma_A)\colon \pi \equiv_s \pi'$$

Theorem 4. *Let $A \subseteq \mathcal{A}$, and let M' be a submodel of M satisfying $\mathbf{AE}_A$. For each* $\mathbf{sATL}^*_{\mathrm{ir}}$ *formula φ over PV, that refers only to coalitions $\hat{A} \subseteq A$, we have: $M, \iota \models_{\mathrm{ir}} \varphi$ iff $M', \iota' \models_{\mathrm{ir}} \varphi$.*

Proof. Proof by induction on the structure of φ. We show the case $\varphi = \langle\langle \hat{A} \rangle\rangle \gamma$. The cases for $\neg, \wedge$ are straightforward.

Note that $out_{M'}(\iota, \sigma_{\hat{A}}) \subseteq out_M(\iota, \sigma_{\hat{A}})$, which together with the condition $\mathbf{AE}_A$ implies that the sets $out_M(\iota, \sigma_{\hat{A}})$ and $out_{M'}(\iota, \sigma_{\hat{A}})$ are stuttering path equivalent. So, the thesis follows from Theorem 1. □

Thus, we have proved that the structural conditions **AE-CF** and $\mathbf{AE}_A$ are sufficient to obtain correct reductions with and without fairness (Theorems 3 and 4). We will discuss algorithms that generate such reduced models in Section 6.

6. Algorithms for Partial Order Reduction

As mentioned above, the idea of model checking with POR is to reduce the size of models while preserving satisfaction for a class of formulas. Traditionally, the reduction algorithm is based either on depth-first-search (DFS, see (Gerth et al., 1999)), or on double-depth-first-search (DDFS (Courcoubetis et al., 1992)). In this paper, we use the former approach.

6.1 DFS Algorithm

DFS is used to search states and transitions that will make up the reduced model by exploring systematically the tree of possible computations, and selecting only some of the possible states and transitions. In the following, the stack represents a path $\pi = g_0\alpha_0 g_1\alpha_1 \cdots g_n$ that is currently being visited. For the top element of the stack g_n, the following three operations are computed in a loop:

1. Identify the set $en(g_n) \subseteq Evt$ of enabled events.
2. Heuristically select a subset $E(g_n) \subseteq en(g_n)$ of possible events (see Section 6.2).
3. For any event $\alpha \in E(g_n)$, compute the successor state g' such that $g_n \xrightarrow{\alpha} g'$, and add g' to the stack thereby generating the path $\pi' = g_0\alpha_0 g_1\alpha_1 \cdots g_n\alpha g'$. Recursively proceed to explore the submodel originating at g' by means of the present algorithm, beginning at step 1.
4. Remove g_n from the stack.

The algorithm begins with the stack comprising of the initial state of $M = IIS(S)$, and terminates when the stack is empty. Note that the model generated by the algorithm must be a submodel of M. Moreover, it is generated *directly from the AMAS S, without ever generating the full model M*. Finally, the size of the reduced model crucially depends on the ratio $E(g)/en(g)$. The choice of $E(g)$ is discussed in the next subsection.

6.2 Heuristics for $\text{sATL}^*_{\text{irF}}$ and Subsets of sATL_{ir}

Let $A \subseteq \mathcal{A}$. The conditions **C1** – **C3** below, inspired by Clarke et al. (1999), define conditions for selection of $E(g) \subseteq en(g)$ in the algorithm of Sect. 6.1.

C1 Along each path π in M that starts at g, each event that is dependent on an event in $E(g)$ cannot be executed in π unless an event in $E(g)$ is executed first in π. Formally, $\forall\pi \in \Pi_M(g)$ such that $\pi = g_0\alpha_0 g_1\alpha_1 \ldots$ with $g_0 = g$, and $\forall\alpha' \in Evt$ such that $(\alpha', \alpha'') \notin I_A$ for some $\alpha'' \in E(g)$, if $\alpha_i = \alpha'$ for some $i \geq 0$, then $\alpha_j \in E(g)$ for some $j < i$.

C2 If $E(g) \neq en(g)$, then $E(g) \subseteq Invis_A$.

C3 For every cycle in M' 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.

If more than one set of events satisfies the above conditions, any of them can be selected as the actual $E(g)$.

Theorem 5. *Let $M = IIS(S)$, and $M' \subseteq M$ be the reduced model generated by DFS with the choice of $E(g')$ for $g' \in St'$ given by conditions* **C1, C3** *and the independence relation I_A, where $A = \emptyset$. Then, M' satisfies* ***AE-CF****.*

Proof. See (Peled, 1996a, Theorem 3.3). □

Theorem 6. *Let $A \subseteq \mathcal{A}$, $M = IIS(S)$, and $M' \subseteq M$ be the reduced model generated by DFS with the choice of $E(g')$ for $g' \in St'$ given by conditions* **C1, C2, C3** *and the independence relation I_A. Then, M' satisfies $\boldsymbol{AE}_A$.*

Proof. Although the setting is slightly different, it can be shown similarly to Clarke et al. (1999, Theorem 12) that the conditions **C1, C2, C3** guarantee that the models M and M' are stuttering path equivalent. More precisely, for each path $\pi = g_0\alpha_0 g_1\alpha_1 \cdots$ with $g_0 = \iota$ in M there is a stuttering equivalent path $\pi' = g'_0\alpha'_0 g'_1\alpha'_1 \cdots$ with $g'_0 = \iota$ in M' such that $Evt(\pi)|_{Vis_A} = Evt(\pi')|_{Vis_A}$, i.e., π and π' have the same maximal sequence of visible events for A.

To show that M' satisfies $\mathbf{AE}_A$, consider an ir-joint strategy σ_A and $\pi \in out_M(\iota, \sigma_A)$. Since $M \equiv_s M'$, we have that there is $\pi' \in \Pi_{M'}(\iota)$ such that $\pi \equiv_s \pi'$ and $Evt(\pi)|_{Vis_A} = Evt(\pi')|_{Vis_A}$. Since $Evt_i \subseteq Vis_A$ for each $i \in A$, the same sequence of events of each Evt_i is executed in π and π'. Thus, by the generalization of Lemma 2 to ir-joint strategies we get $\pi' \in out_M(\iota, \sigma_A)$. So, by Lemma 1 we have $\pi' \in out_M(\iota, \sigma_A)$. □

Thus, we have obtained a general method of POR for fragments of $\mathbf{ATL}^*$ with imperfect information. The method is in fact a reformulation of the reduction for $\mathbf{LTL}_{-\mathbf{X}}$. This has at least two welcome implications. First, the actual reductions are likely to be substantial – much more than one would expect with the expressivity of $\mathbf{sATL}^*$. Secondly, one can reuse or adapt existing algorithms and tools performing reductions for $\mathbf{LTL}_{-\mathbf{X}}$. Algorithms generating reduced models, in which the choice of $E(g)$ is given by **C1, C2, C3** or **C1, C3** can be found for instance in the works of Peled (1996a, 1993), Clarke et al. (1999), Gerth et al. (1999), Penczek et al. (2000), Lomuscio et al. (2010b).

6.3 Bad News for Agents with Perfect Information

Here, we briefly show that the adaptation of $\mathbf{LTL}_{-\mathbf{X}}$ reduction, proposed in this paper, does not work for $\mathbf{sATL}^*$ with memoryless perfect information. We begin with a counterexample to Lemma 3 which was essential to our formal construction (Example 5). Then, we show that the whole method does not preserve formulas of $\mathbf{sATL}^*_{\mathrm{Ir}}$ (Example 6).

Example 5. *Consider the MAS composed of two agents $\{1,2\}$ such that: $L_1 = \{l_1^1, l_1^2\}$, $L_2 = \{l_2^1, l_2^2\}$, $Evt_1 = \{\epsilon, a\}$, $Evt_2 = \{\epsilon, b\}$, $P_1(l_1^1) = \{a, \epsilon\}$, $P_1(l_1^2) = \{\epsilon\}$, $P_2(l_2^1) = \{b\}$, $P_2(l_2^2) = \{\epsilon\}$, and $T_1(l_1^1, a) = l_1^2$, $T_2(l_2^1, b) = l_2^2$.*

Define an Ir*-strategy $\sigma_{\{1,2\}}$ as follows: $\sigma_1(l_1^1, l_2^1) = a$, $\sigma_1(l_1^1, l_2^2) = \sigma_1(l_1^2, l_2^2) = \epsilon$; $\sigma_2(l_1^1, l_2^1) = \sigma_2(l_1^2, l_2^1) = b$, $\sigma_2(l_1^2, l_2^2) = \epsilon$. It is easy to see that $out((l_1^1, l_2^1), \sigma_{\{1,2\}})$ is not trace complete. Note 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 $ba(\epsilon)^\omega$.*

Example 6. *Consider formula $\langle\langle c\rangle\rangle(\mathrm{F}\ \mathsf{in}_1 \wedge \mathrm{F}\ \mathsf{in}_2)$, interpreted with the* Ir *semantics. Clearly, the formula holds in the TGC model in Figure 2 (left), but not in the reduced model in Figure 2 (right).*

7. Experimental Evaluation

In this section, we demonstrate the efficiency of partial order reduction for strategic abilities under imperfect information. We already pointed out that the main strength of our results

lies in the fact that our reduction for **sATL*** is essentially the same as the well-known reduction for **LTL** (Peled, 1993, 1996a, 1998; Clarke et al., 1999). This has two important consequences. First, the reduction for **sATL*** is based on a weak notion of model equivalence – much weaker than the equivalences for **CTL*** and full **ATL***. Hence, it allows for relatively coarse equivalence classes, and produces relatively small models. Secondly, it can be done by means of an existing POR implementation for **LTL**. That is, we can take an existing tool off the shelf, and use it to do reductions for strategic abilities. This way, we do not only avoid the burden of implementing the algorithms and heuristics on our own. We also benefit from the 20 years of development and optimization of model checking tools for linear time logic, which resulted in remarkably efficient partial order reduction (Peled, 1998; Lomuscio et al., 2010b; Meulen & Pecheur, 2011).

To evaluate the gains that the technique produces for asynchronous multi-agent systems, we have used three benchmarks: classical Train-Gate-Controller, Pipeline, and Asynchronous Simple Voting. We present the benchmarks in more detail in Section 7.1. We have implemented the corresponding asynchronous MAS in Promela, the modeling language of the Spin model checker (Holzmannn, 1997). Then, we used the partial order reduction functionality of Spin to produce reduced models, and compared their sizes to the full models. The results are presented in Section 7.2. Again, the choice to use an existing, well-known **LTL** model checker was made on purpose, to further demonstrate our claim that the existing techniques and tools can be adapted for strategic operators in multi-agent systems. The implementation of the partial order reduction algorithm in Spin is detailed by Holzmann and Peled (1995).

7.1 Benchmarks

We ran our experiments for three classes of scalable asynchronous multi-agent systems: Train-Gate-Controller, Pipeline, and Asynchronous Simple Voting.

Train-Gate-Controller (TGC). The TGC benchmark, inspired by Alur et al. (1993, 1998), Hoek and Wooldridge (2002), was already presented in Section 2 and used as the running example in the subsequent sections. In particular, the asynchronous MAS for TGC was presented in Example 1, and its unfolding to an interleaved interpreted system was shown in Example 2. We recall that the variant of TGC with n trains is denoted by TGC_n.

Pipeline. This is a benchmark adapted from Peled (1993), and featuring a sequence of n processes $p_1, \ldots, p_n$, each of length m (except for p_1 and p_n, whose length is fixed at 2). Resembling a physical pipeline, they are arranged so that an output transition of p_i is always synchronized with the input transition of p_{i+1}.

Asynchronous Simple Voting (ASV). Finally, we propose a new benchmark, inspired by the simple model of voting and coercion from Jamroga et al. (2017). There are n voters $v_1, \ldots, v_n$ choosing between k candidates $1, \ldots, k$. Additionally, a coercer agent c may attempt to coerce a voter into voting for a specific candidate. The voter has then the choice between providing the coercer with a proof (i.e., ballot receipt) of how they voted, or refusing to do so.

We denote an AMAS for the scenario by $ASV_{n,k}$. The simplest variant, for $n = 2$ voters and $k = 2$ candidates, is presented in Figure 3. The set of propositional variables is

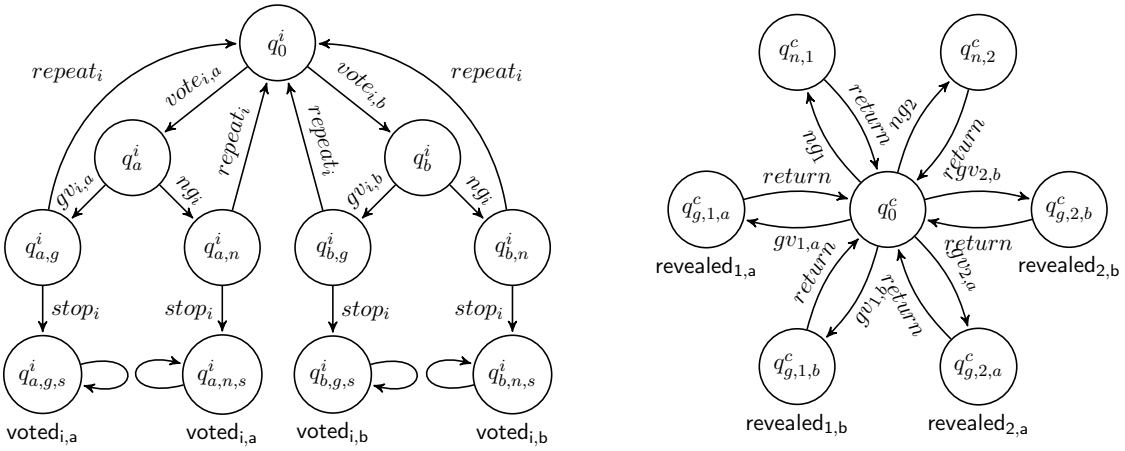

Figure 3: Voter v_i (left) and coercer c (right) in the ASV benchmark.

$\mathcal{PV} = \{\mathsf{voted_{1,1}}, \ldots, \mathsf{voted_{n,k}}, \mathsf{revealed_{1,1}}, \ldots, \mathsf{revealed_{n,k}}\}$. Proposition $\mathsf{voted_{i,j}}$ denotes that v_i voted for the j-th candidate, while $\mathsf{revealed_{i,j}}$ denotes that v_i additionally gave the coercer the proof of having voted for j.

7.2 Experimental Results

TGC. The TGC benchmark is an example of an agent system excellently suited for partial order reduction. As the input formula, we have used $\phi_1 = \langle\langle c \rangle\rangle \mathrm{G}\, \neg(\bigvee_{1 \leq i \neq j \leq n}(\mathsf{in_i} \wedge \mathsf{in_j}))$, saying that the controller can ensure that different trains are never simultaneously in the tunnel. Based on the estimates from Lomuscio et al. (2010a, 2010b), Jamroga et al. (2018), the size of the full state space is $|St_{IIS(TGC_n)}| = \mathcal{O}(2^{n+1})$, while the size of the reduced model is $|St_{M'_n}| = \mathcal{O}(n)$. This is confirmed by our experimental results, presented in Figure 4, which show that the reduction is indeed very significant.

Pipeline. The results for Pipeline are presented in Figure 5. The input formula is $\phi_2 = \langle\langle p_i \rangle\rangle \mathrm{G}\, (\bigwedge_{1 \leq i \leq n}(\neg \mathsf{in_i}\, \mathrm{U}\, (\mathsf{out_1} \wedge \cdots \wedge \mathsf{out_{i-1}} \wedge \mathsf{in_i})))$, where p_i is the i-th process in the pipeline and propositions $\mathsf{in_i}, \mathsf{out_i}$ denote, respectively, that p_i has started processing and that it has delivered its output. Thus, ϕ_2 expresses that no process in the pipeline will start operating until the output from every previous one has been delivered.

While not exponential as in the case of TGC, the reduction can be considered very substantial as it exceeds 90% for larger instances. Moreover, its effectiveness significantly increases not only with the number of processes (n), but also with their length (m).

ASV. The reduction for Asynchronous Simple Voting was done in the context of formula $\phi_3 = \langle\langle v_i \rangle\rangle \mathrm{F}\, (\mathsf{voted_{i,a}} \wedge \neg \mathsf{revealed_{i,a}} \wedge \mathsf{revealed_{i,b}})$, expressing that voter v_i has a strategy to eventually have voted for candidate a, and revealed a receipt for a b vote instead. The set of relevant propositions is now $\mathcal{PV} = \{\mathsf{voted_{i,a}}, \mathsf{revealed_{i,a}}, \mathsf{revealed_{i,b}}\}$. In addition to the number of voters n, we also scaled on the number of candidates k. The experimental results for $k = 2$, $k = 3$ and $k = 4$ candidates are presented in Figure 6. In each instance, we consider two cases, i.e., the reductions obtained with and without the concurrency fairness assumption.

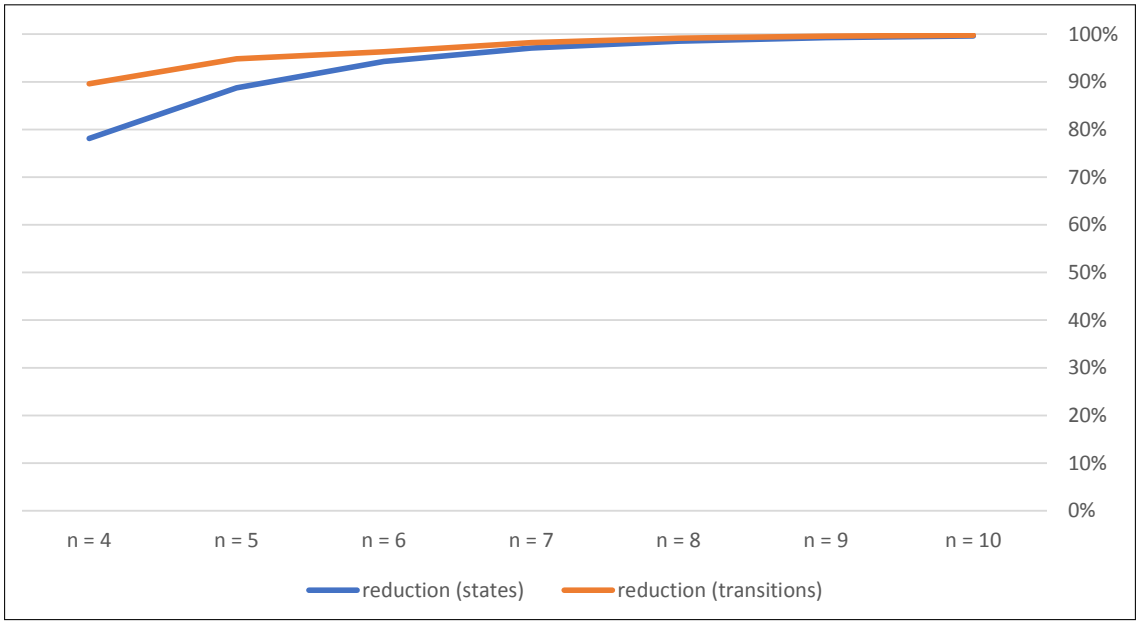

Figure 4: Effectiveness of reduction for the TGC benchmark with n trains.

The effectiveness of the reduction, while also not exponential, is nevertheless significant. For example, for $n = 7$ (i.e., 7 voters + 1 coercer), we got 62% reduction of the state space and up to 70% reduction of the transition space. Moreover, the reduction becomes more effective as the size of the MAS increases. Note also that, similarly to the previous benchmarks, it produces larger gains w.r.t. the number of transitions rather than states, i.e., it handles better the more important factor in the complexity of **ATL** model checking.

Finally, we observe that the choice of the semantics changes the output of the reduction in case of ASV, unlike the previous benchmarks. In particular, transition ng_i of voter v_i does not modify the propositions in $\mathcal{PV}$, and thus can potentially be omitted in the reduced model for $\mathbf{sATL}^*_{\mathrm{ir}}$. On the other hand, it is visible under the concurrency-fairness assumption when we want to verify the abilities of the coercer (as in formula ϕ_3). Thus, the reduction for $\mathbf{sATL}^*_{\mathrm{irF}}$ can be less effective.

7.3 Summary

The results presented in this section demonstrate that partial order reduction can provide significant gains in terms of the size of the model that would be used as the input to verification. In some cases, like in the classical TGC benchmark, POR produces exponentially smaller state- and transition-spaces. Of course, such optimistic results are by no means guaranteed. For many asynchronous MAS, the reduction may remove a smaller fraction of states, as demonstrated by the other two benchmarks. We note, however, that the reductions obtained for Pipeline and ASV are also substantial: certainly better than linear, and possibly close to quadratic w.r.t. the number of agents. This seems very promising, especially considering the fact that $\mathbf{sATL}^*_{\mathrm{ir}}$ model checking is **NP**-hard *in the size of the model*, and all the attempts at practical algorithms so far run in exponential time. Thus,

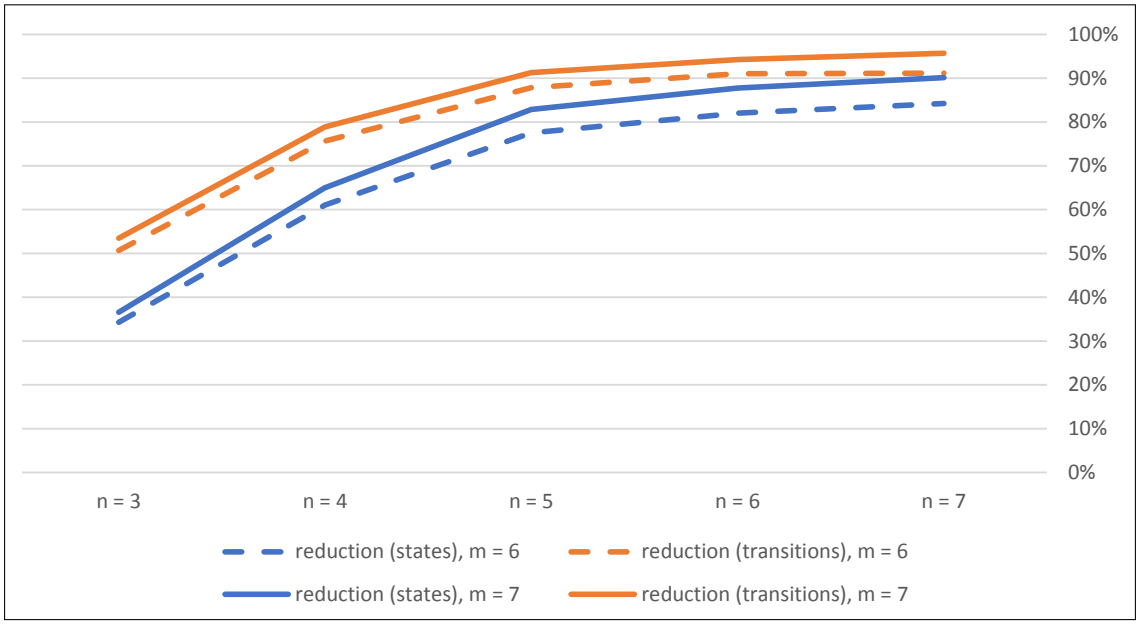

Figure 5: Effectiveness of reduction for the Pipeline benchmark with n processes, each of length m.

even a polynomial reduction of the state/transition space is likely to produce an exponential improvement of the performance.

8. Conclusions and Future Work

Many important properties of multi-agent systems are underpinned by the ability of some agents (or groups) to achieve a given goal. In this paper, we propose a general semantics of strategic ability for asynchronous MAS, and study the model checking problem for relevant subsets of alternating-time temporal logic. We concentrate on imperfect information strategies, and consider two semantic variants: one looking at *all* the infinite executions of strategies, and the other taking into account only the *fair* execution paths.

The theoretical complexity results follow the same pattern as those for synchronous MAS, though proving them required careful treatment in some cases. Consequently, model checking of strategic abilities under imperfect information for asynchronous systems is as hard as in the synchronous case. This makes model reductions essential for practical verification. The most important result of this paper consists in showing that the partial order reduction for $\mathbf{LTL}_{-\mathbf{X}}$ can be almost directly applied to $\mathbf{ATL}_{\mathrm{ir}}$ without nested strategic modalities. The importance of the result stems from the fact that $\mathbf{LTL}_{-\mathbf{X}}$ has relatively weak distinguishing power, and therefore admits strong reductions, clustering paths into relatively few equivalence classes.

Interestingly, it turns out that the scheme does *not* work for $\mathbf{ATL}^*$ with 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

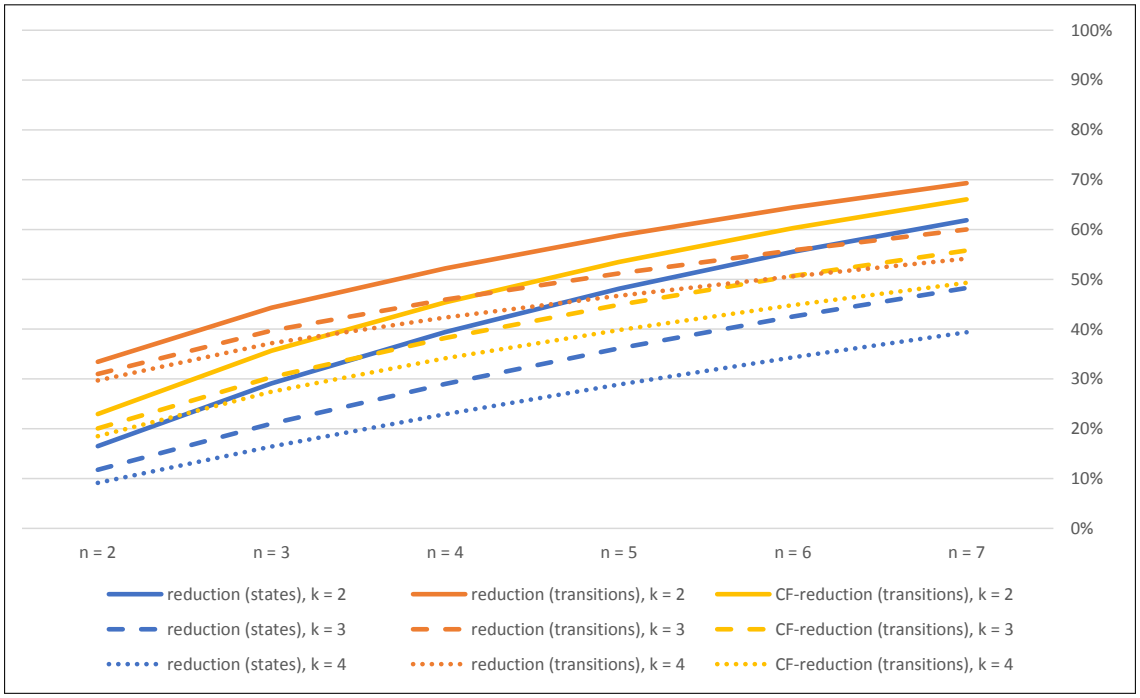

Figure 6: Effectiveness of reduction for the ASV benchmark with n voters and k candidates.

identify an aspect of verification that might be in favour of imperfect information strategies in some contexts.

The ideas presented in this paper open many exciting paths for future research. We will have a closer look at some alternative semantics for $\mathbf{ATL}_{\mathrm{ir}}$ in asynchronous MAS, including the "deadlock-friendly" semantics and the one based on "subjective" ability. Extending the semantics by allowing for nondeterministic strategies of players is another possibility. We suspect that, unlike for synchronous systems, allowing for nondeterministic choices in strategies might make a difference in the semantics. We also plan to extend our method to a larger subset of $\mathbf{ATL}^*$ specifications, a subset of Strategy Logic (Berthon et al., 2017a), and to $\mathbf{sATL}^*$ with epistemic operators using possibly techniques reported by Cermák et al. (2014). Further experimental evaluation of the reductions on randomly generated models is also on the list. Adapting the POR scheme to combinations of strategic and epistemic modalities is another interesting path for future work. Finally, we would like to investigate if our partial order reduction scheme can be combined with the bisimulation-based reduction for $\mathbf{ATL}_{\mathrm{ir}}$, proposed recently by Belardinelli et al. (2017b).

Acknowledgements

We thank the anonymous reviewers for insightful comments. W. Jamroga and W. Penczek acknowledge the support of the National Centre for Research and Development, Poland (NCBR), and the Luxembourg National Research Fund (FNR), under the PolLux/FNR-CORE projects VoteVerif (POLLUX-IV/1/2016) and STV (POLLUX-VII/1/2019).

W. Penczek and T. Sidoruk acknowledge also the support of the French National Centre for Scientific Research (CNRS), and the Polish Academy of Sciences (PAN), under the CNRS/PAN project PARTIES.

References

Abdulla, P. A., Aronis, S., Jonsson, B., & Sagonas, K. F. (2014). Optimal dynamic partial order reduction. In *Proceedings of the 41st Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL '14, San Diego, CA, USA, January 20-21, 2014*, pp. 373–384. ACM.

Alur, R., de Alfaro, L., Grossu, R., Henzinger, T., Kang, M., Kirsch, C., Majumdar, R., Mang, F., & Wang, B.-Y. (2001). jMocha: A model-checking tool that exploits design structure. In *Proceedings of the 23rd International Conference on Software Engineering, ICSE '01, Toronto, ON, Canada, May 12-19, 2001*, pp. 835–836. IEEE Computer Society Press.

Alur, R., Henzinger, T., Mang, F., Qadeer, S., Rajamani, S., & Tasiran, S. (1998). MOCHA: Modularity in model checking. In *Proceedings of the 10th International Conference on Computer Aided Verification, CAV '98, Vancouver, BC, Canada, June 28 - July 2, 1998*, Vol. 1427 of *Lecture Notes in Computer Science*, pp. 521–525. Springer.

Alur, R., & Henzinger, T. A. (1999). Reactive modules. *Formal Methods in System Design*, *15*(1), 7–48.

Alur, R., Henzinger, T. A., & Kupferman, O. (1997). Alternating-time Temporal Logic. In *Proceedings of the 38th Annual Symposium on Foundations of Computer Science, FOCS '97, Miami Beach, FL, USA, October 19-22, 1997*, pp. 100–109. IEEE Computer Society Press.

Alur, R., Henzinger, T. A., & Kupferman, O. (2002). Alternating-time Temporal Logic. *Journal of the ACM*, *49*, 672–713.

Alur, R., Henzinger, T. A., & Vardi, M. Y. (1993). Parametric real-time reasoning. In *Proceedings of the 25th Annual ACM Symposium on Theory of Computing, STOC '93, San Diego, CA, USA, May 16-18, 1993*, pp. 592–601. ACM.

Baier, C., & Katoen, J.-P. (2008). *Principles of Model Checking*. MIT Press.

Belardinelli, F., Lomuscio, A., Murano, A., & Rubin, S. (2017a). Verification of broadcasting multi-agent systems against an epistemic strategy logic. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pp. 91–97. AAAI Press.

Belardinelli, F., Condurache, R., Dima, C., Jamroga, W., & Jones, A. V. (2017b). Bisimulations for verifying strategic abilities with an application to ThreeBallot. In *Proceedings of the 16th Conference on Autonomous Agents and Multi-agent Systems, AAMAS 2017, São Paulo, Brazil, May 8-12, 2017*, pp. 1286–1295. ACM.

Belardinelli, F., Lomuscio, A., Murano, A., & Rubin, S. (2017c). Verification of multi-agent systems with imperfect information and public actions. In *Proceedings of the*

16th Conference on Autonomous Agents and Multi-agent Systems, AAMAS 2017, São Paulo, Brazil, May 8-12, 2017, pp. 1268–1276. ACM.

Berthon, R., Maubert, B., Murano, A., Rubin, S., & Vardi, M. Y. (2017a). Strategy logic with imperfect information. In *Proceedings of the 32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017, Reykjavik, Iceland, June 20-23, 2017*, pp. 1–12. IEEE.

Berthon, R., Maubert, B., & Murano, A. (2017b). Decidability results for ATL* with imperfect information and perfect recall. In *Proceedings of the 16th Conference on Autonomous Agents and Multi-agent Systems, AAMAS 2017, São Paulo, Brazil, May 8-12, 2017*, pp. 1250–1258. ACM.

Bulling, N., Dix, J., & Jamroga, W. (2010). Model checking logics of strategic ability: Complexity. In Dastani, M., Hindriks, K., & Meyer, J.-J. (Eds.), *Specification and Verification of Multi-Agent Systems*, pp. 125–159. Springer.

Bulling, N., & Jamroga, W. (2014). 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.

Bulling, N., & Jamroga, W. (2011). Alternating epistemic mu-calculus. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence, IJCAI 2011, Barcelona, Catalonia, Spain, July 16-22, 2011*, pp. 109–114. AAAI Press.

Busard, S., Pecheur, C., Qu, H., & Raimondi, F. (2014). Improving the model checking of strategies under partial observability and fairness constraints. In *Formal Methods and Software Engineering*, Vol. 8829 of *Lecture Notes in Computer Science*, pp. 27–42. Springer.

Busard, S., Pecheur, C., Qu, H., & Raimondi, F. (2015). Reasoning about memoryless strategies under partial observability and unconditional fairness constraints. *Information and Computation*, *242*, 128–156.

Cermák, P., Lomuscio, A., Mogavero, F., & Murano, A. (2014). MCMAS-SLK: A model checker for the verification of strategy logic specifications. In *Proceedings of the 26th International Conference on Computer Aided Verification, CAV 2014, Vienna, Austria, July 18-22, 2014*, Vol. 8559 of *Lecture Notes in Computer Science*, pp. 525–532. Springer.

Cermák, P., Lomuscio, A., & Murano, A. (2015). Verifying and synthesising multi-agent systems against one-goal strategy logic specifications. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence, AAAI-15, January 25-30, 2015, Austin, TX, USA*, pp. 2038–2044.

Chatterjee, K., Pavlogiannis, A., Sinha, N., & Vaidya, K. (2016). Data-centric dynamic partial order reduction. *CoRR*, *abs/1610.01188*.

Chen, T., Forejt, V., Kwiatkowska, M., Parker, D., & Simaitis, A. (2013). PRISM-games: A model checker for stochastic multi-player games. In *Proceedings of the 19th International Conference on Tools and Algorithms for the Construction and Analysis of Systems, TACAS 2013, Rome, Italy, March 16-24, 2013*, Vol. 7795 of *Lecture Notes in Computer Science*, pp. 185–191. Springer.

Clarke, E. M., Grumberg, O., & Peled, D. A. (1999). *Model Checking*. MIT Press.

Clarke, E., & Emerson, E. (1981). Design and synthesis of synchronization skeletons using branching time temporal logic. In *Proceedings of Logics of Programs Workshop*, Vol. 131 of *Lecture Notes in Computer Science*, pp. 52–71.

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

Dastani, M., & Jamroga, W. (2010). Reasoning about strategies of multi-agent programs. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2010, Toronto, ON, Canada, May 10-14, 2010*, pp. 997–1004. IFAAMAS.

Dima, C., Maubert, B., & Pinchinat, S. (2014). The expressive power of epistemic μ-calculus. *CoRR*, *abs/1407.5166*.

Dima, C., Maubert, B., & Pinchinat, S. (2015). Relating paths in transition systems: The fall of the modal mu-calculus. In *Proceedings of the 40th International Symposium on Mathematical Foundations of Computer Science, MFCS 2015, Milan, Italy, August 24-28, 2015*, Vol. 9234 of *Lecture Notes in Computer Science*, pp. 179–191. Springer.

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

Fagin, R., Halpern, J. Y., Moses, Y., & Vardi, M. Y. (1995). *Reasoning about Knowledge*. MIT Press.

Flanagan, C., & Godefroid, P. (2005). 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, CA, USA, January 12-14, 2005*, pp. 110–121. ACM.

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

Godefroid, P. (1991). Using partial orders to improve automatic verification methods. In *Proceedings of the 2nd International Conference on Computer Aided Verification, CAV '90, New Brunswick, NJ, USA, June 18-21, 1990*, pp. 321–340.

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

Guelev, D., Dima, C., & Enea, C. (2011). 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.

Hoek, W., & Wooldridge, M. (2002). Tractable multiagent planning for epistemic goals. In *Proceedings of the 1st International Joint Conference on Autonomous Agents and Multi-agent Systems, AAMAS 2002, Bologna, Italy, July 15-19, 2002*, pp. 1167–1174. ACM.

Holzmann, G., & Peled, D. (1995). An improvement in formal verification. In *Proceedings of the 7th IFIP WG 6.1 International Conference on Formal Description Techniques, FORTE 1994, Berne, Switzerland, October 4-7, 1994*, pp. 197–211. Springer.

Holzmannn, G. J. (1997). The model checker SPIN. *IEEE Transactions on Software Engineering*, *23*(5), 279–295.

Huang, X., & van der Meyden, R. (2014). Symbolic model checking epistemic strategy logic. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence, AAAI-14, Quebec City, QC, Canada, July 27-31, 2014*, Vol. 2, pp. 1426–1432. AAAI Press.

Jamroga, W. (2003). Some remarks on alternating temporal epistemic logic. In *Proceedings of the 1st Workshop on Formal Approaches to Multi-Agent Systems, FAMAS 2003, Warsaw, Poland, April 5-13, 2003*, pp. 133–140. University of Warsaw.

Jamroga, W. (2015). *Logical Methods for Specification and Verification of Multi-Agent Systems*. ICS PAS Publishing House.

Jamroga, W., & Dix, J. (2006). Model checking ATL$_{ir}$ is indeed Δ_2^P-complete. In *Proceedings of the 4th European Workshop on Multi-Agent Systems, EUMAS 2006, Lisbon, Portugal, December 14-15, 2006*. CEUR-WS.org.

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

Jamroga, W., & Agotnes, T. (2007). Modular interpreted systems. In *Proceedings of the 6th International Joint Conference on Autonomous Agents and Multi-agent Systems, AAMAS 2007, Honolulu, HI, USA, May 14-18, 2007*, pp. 897–904. IFAAMAS.

Jamroga, W., Knapik, M., & Kurpiewski, D. (2017). Fixpoint approximation of strategic abilities under imperfect information. In *Proceedings of the 16th Conference on Autonomous Agents and Multi-agent Systems, AAMAS 2017, São Paulo, Brazil, May 8-12, 2017*, pp. 1241–1249. ACM.

Jamroga, W., Knapik, M., Kurpiewski, D., & Mikulski, Ł. (2019). Approximate verification of strategic abilities under imperfect information. *Artificial Intelligence*, *277*.

Jamroga, W., Penczek, W., Dembiński, P., & Mazurkiewicz, A. W. (2018). Towards partial order reductions for strategic ability. In *Proceedings of the 17th International Conference on Autonomous Agents and Multi-agent Systems, AAMAS 2018, Stockholm, Sweden, July 10-15, 2018*, pp. 156–165. ACM.

Kacprzak, M., & Penczek, W. (2004). Unbounded model checking for alternating-time temporal logic. In *Proceedings of the 3rd International Joint Conference on Autonomous Agents and Multi-agent Systems, AAMAS 2004, 19-23 August 2004, New York, NY, USA*, pp. 646–653. IEEE Computer Society.

Kahlon, V., Wang, C., & Gupta, A. (2009). Monotonic partial order reduction: An optimal symbolic partial order reduction technique. In *Proceedings of the 21st International Conference on Computer Aided Verification, CAV 2009, Grenoble, France, June 26 - July 2, 2009*, pp. 398–413. Springer.

Kokkarinen, I., Peled, D., & Valmari, A. (1997). Relaxed visibility enhances partial order reductions. In *Proceedings of the 9th International Conference on Computer Aided Verification, CAV '97, Haifa, Israel, June 22-25, 1997*, pp. 328–340. Springer-Verlag.

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

Kristensen, L. M., & Valmari, A. (2000). 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, Aarhus, Denmark, June 26-30, 2000*, pp. 282–302. Springer-Verlag.

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

Kurpiewski, D., Jamroga, W., & Knapik, M. (2019a). STV: model checking for strategies under imperfect information. In *Proceedings of the 18th International Conference on Autonomous Agents and Multi-agent Systems, AAMAS 2019, Montreal, QC, Canada, May 13-17, 2019*, pp. 2372–2374. IFAAMAS.

Kurpiewski, D., Knapik, M., & Jamroga, W. (2019b). On domination and control in strategic ability. In *Proceedings of the 18th International Conference on Autonomous Agents and Multi-agent Systems, AAMAS 2019, Montreal, QC, Canada, May 13-17, 2019*, pp. 197–205. IFAAMAS.

Laroussinie, F., Markey, N., & Oreiby, G. (2008). On the expressiveness and complexity of ATL. *Logical Methods in Computer Science*, *4*(2), 1–25.

Lomuscio, A., Qu, H., & Raimondi, F. (2009). MCMAS: A model checker for the verification multi-agent systems. In *Proceedings of the 21st International Conference on Computer Aided Verification, CAV 2009, Grenoble, France, June 26 - July 2, 2009*, pp. 682–688. Springer.

Lomuscio, A., & Raimondi, F. (2006). MCMAS: A model checker for multi-agent systems. In *Proceedings of the 12th International Conference on Tools and Algorithms for the Construction and Analysis of Systems, TACAS 2006, Vienna, Austria, March 25 - April 2, 2006*, pp. 450–454. Springer.

Lomuscio, A., & Sergot, M. (2003). Deontic interpreted systems. *Studia Logica*, *75*(1), 63–92.

Lomuscio, A., Penczek, W., & Qu, H. (2010a). Partial order reductions for model checking temporal epistemic logics over interleaved multi-agent systems. In *Proceedings of the 9th International Conference on Autonomous Agents and Multi-agent Systems, AAMAS 2010, Toronto, ON, Canada, May 10-14, 2010*, Vol. 1, pp. 659–666. IFAAMAS.

Lomuscio, A., Penczek, W., & Qu, H. (2010b). Partial order reductions for model checking temporal-epistemic logics over interleaved multi-agent systems. *Fundamenta Informaticae*, *101*(1-2), 71–90.

Lomuscio, A., Qu, H., & Raimondi, F. (2017). Mcmas: an open-source model checker for the verification of multi-agent systems. *International Journal on Software Tools for Technology Transfer*, *19*(1), 9–30.

Lomuscio, A., & Raimondi, F. (2006). Model checking knowledge, strategies, and games in multi-agent systems. In *Proceedings of the 5th International Joint Conference on*

Autonomous Agents and Multiagent Systems, AAMAS 2006, Hakodate, Japan, May 8-12, 2006, pp. 161–168. ACM.

Malvone, V., Murano, A., & Sorrentino, L. (2017). Hiding actions in multi-player games. In *Proceedings of the 16th Conference on Autonomous Agents and Multi-agent Systems, AAMAS 2017, São Paulo, Brazil, May 8-12, 2017*, pp. 1205–1213. ACM.

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

Mazurkiewicz, A. (1986). Trace theory. In *Advances in Petri Nets 1986*, Vol. 255 of *LNCS*, pp. 279–324. Springer-Verlag.

Mazurkiewicz, A. (1988). Basic notions of trace theory. In *Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency*, Vol. 354 of *LNCS*, pp. 285–363. Springer-Verlag.

Meski, A., Penczek, W., Szreter, M., Woźna-Szcześniak, B., & Zbrzezny, A. (2014). BDD-versus SAT-based bounded model checking for the existential fragment of linear temporal logic with knowledge: algorithms and their performance. *Autonomous Agents and Multi-Agent Systems*, *28*(4), 558–604.

Meulen, J. V., & Pecheur, C. (2011). Combining partial-order reduction and symbolic model checking to verify LTL properties. In *Proceedings of the 3rd International Symposium on NASA Formal Methods, NFM 2011, Pasadena, CA, USA, April 18-20, 2011*, pp. 406–421. Springer.

Peled, D. (1993). All from one, one for all: On model checking using representatives. In *Proceedings of the 5th International Conference on Computer Aided Verification, CAV '93, Elounda, Greece, June 28 - July 1, 1993*, pp. 409–423. Springer-Verlag.

Peled, D. (1996a). Combining partial order reductions with on-the-fly model-checking. *Formal Methods in System Design*, *8*(1), 39–64.

Peled, D. (1996b). Partial order reductions: Model checking using representatives. In *Proceedings of the 21st International Symposium on Mathematical Foundations of Computer Science, MFCS '96, Cracow, Poland, September 2-6, 1996*, pp. 93–112. Springer-Verlag.

Peled, D. (1998). Ten years of partial-order reductions. In *Proceedings of the 10th International Conference on Computer Aided Verification, CAV '98, Vancouver, BC, Canada, June 28 - July 2, 1998*, pp. 17–28. Springer-Verlag.

Penczek, W., Szreter, M., Gerth, R., & Kuiper, R. (2000). Improving partial order reductions for universal branching time properties. *Fundamenta Informaticae*, *43*, 245–267.

Pilecki, J., Bednarczyk, M., & Jamroga, W. (2014). Synthesis and verification of uniform strategies for multi-agent systems. In *Proceedings of the 15th International Workshop on Computational Logic in Multi-Agent Systems, CLIMA XV, Prague, Czech Republic, August 18-19, 2014*, pp. 166–182. Springer.

Priese, L. (1983). Automata and concurrency. *Theoretical Computer Science*, *25*(3), 221 – 265.

Puchala, B. (2010). Asynchronous omega-regular games with partial information. In *Proceedings of the 35th International Symposium on Mathematical Foundations of Computer Science, MFCS 2010, Brno, Czech Republic, August 23-27, 2010*, pp. 592–603. Springer.

Raimondi, F. (2006). *Model Checking Multi-agent Systems*. Ph.D. thesis, University College London.

Schnoebelen, P. (2003). The complexity of temporal model checking. In *Proceedings of the 4th Conference on Advances in Modal Logics, AiML 2002, Toulouse, France, September 30 - October 2, 2002*. World Scientific.

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

van der Hoek, W., Lomuscio, A., & Wooldridge, M. J. (2006). On the complexity of practical ATL model checking. In *Proceedings of the 5th International Joint Conference on Autonomous Agents and Multi-agent Systems, AAMAS 2006, Hakodate, Japan, May 8-12, 2006*, pp. 201–208. ACM.