INSTITUT POLYTECHNIQUE DE PARIS

Habilitation à Diriger des Recherches (HDR)

Formal Modeling, Specification, and Verification of Multi-Agent Systems

HDR de l'Institut Polytechnique de Paris
préparée à Télécom Paris

Thèse présentée et soutenue à Palaiseau, le 25/06/2024, par

Vadim MALVONE

Composition du Jury :

Etienne André Professeur, Université Sorbonne Paris Nord	Examinateur
Giuseppe De Giacomo Full professor, Sapienza Universita' di Roma	Rapporteur
Stéphane Demri Senior researcher, LMF, CNRS & ENS Paris-Saclay	Rapporteur
Catalin Dima Professeur, Université Paris-Est Créteil	Examinateur
Costantin Enea Professeur, Ecole Polytechnique, LIX	Examinateur
Andreas Herzig Directeur de Recherches CNRS, Université Paul Sabatier IRIT-LILaC	Rapporteur
François Laroussinie Professeur, l'Université Paris Diderot Paris 7	Examinateur

Contents

Chapter 1

Introduction

The challenge of ensuring system correctness is particularly significant in hardware and software design, especially in safety-critical scenarios. When referring to a safety-critical system, we mean one in which failure is not an option. These systems are typically categorized as follows: safety-critical, where errors can have life-threatening consequences; mission-critical, where unreliability can impact the achievement of objectives; and business-critical, where failure can result in financial losses.

In recent years, there have been several instances of critical systems exhibiting unexpected behavior with relevant consequences. For instance, in February 2020, over 100 flights at Heathrow Airport were disrupted due to a software "glitch". Additionally, in December 2020, Google's services experienced a 45-minute outage because their system could not recover from a storage issue. In March 2020, Finastra, a leading banking software provider, had to take its systems offline following a ransomware attack. This problem is not diminishing, as per Cybersecurity Ventures, the cost of cybercrime alone is projected to reach $10.5 trillion annually by 2025.

To address this issue, several methodologies have been proposed. Among these, model checking, as described in [1, 2], proves to be highly useful. This approach offers a formal-based methodology for modeling systems, specifying properties, and verifying that a system adheres to a given specification. Typically, the mathematical model takes the form of a labeled transition graph, with each node representing a system state, and the edges indicating transitions between states resulting from system execution. In order to specify properties, temporal logics are commonly employed, such as Linear-time Temporal Logic (LTL) [3] and Computation Tree Logic (CTL) [4].

Initially, model checking applications primarily focused on closed systems, characterized by behavior entirely determined by their internal states. However, these model checking techniques designed for closed systems proved to have limited practical utility, as most systems are open and engage in ongoing interactions with other systems. To address this challenge, model checking was extended to Multi-Agent Systems (MAS). In this context, labeled transition graphs are augmented to incorporate actions. This means that the edges between states are labeled according to the actions chosen by the agents participating in the MAS. Concurrent Game Structures (CGS) [5] and Interpreted Systems (IS) [6] are commonly employed in formal verification to model MAS. Regarding the specification, temporal logics have been expanded to incorporate strategic reasoning, such as Alternating-time Temporal Logic (ATL) [5] and Strategy Logic (SL) [7]. To delve into more detail, ATL extends CTL by replacing the existential and universal path quantifiers with strategic modalities represented as $\langle\langle A \rangle\rangle$ and $[\![A]\!]$, where A refers to a set of agents. The existential strategic operator $\langle\langle A \rangle\rangle$ allows us to verify whether there exists a strategy for the coalition A such that, for any action taken by the other agents, it is possible to achieve a strategic objective. Conversely, the universal operator $[\![A]\!]$ is the complement of the existential one. Informally, a strategy can be described as a conditional plan that specifies an action for every situation within a MAS. While ATL is expressive, it has a significant limitation in that it treats strategies only implicitly within the semantics of its strategic operators. This limitation renders the logic less suitable for formalizing crucial solution concepts, such as the Nash Equilibrium. These considerations prompted the development and exploration of Strategy Logic, an extension of LTL, offering a more robust formalism for strategic reasoning. A key aspect of this logic is its treatment of strategies as first-order objects, which can be specified using the existential $\exists x$ and universal $\forall x$ quantifiers. These quantifiers can be respectively interpreted as "there exists a strategy x" and "for all strategies x". Consequently, these plans are not inherently tied to a specific agent, and an explicit binding operator (α, x) enables the association of an agent α with a strategy represented by the variable x.

In relation to the formal model selected to describe the MAS and the logic chosen to specify the property of interest, various algorithms and automaton-based approaches have been proposed [5, 7, 8]. The interesting aspect is that the complexity of model checking can range from polynomial to non-elementary and, in some contexts, even reach undecidability. To discuss these results, it is necessary to introduce two fundamental aspects in MAS verification that impact the complexities of model checking: the memory of strategies and agent visibility.

In MAS, a strategy is generally defined as a function that for each MAS situation returns an action. In particular, we can distinguish between two main types of strategies: memoryless and memoryfull. With memoryless (aka positional or imperfect recall) strategies, the agent does not remember the past but only the current MAS state. In contrast, with memoryfull (aka perfect recall) strategies the agent considers all the MAS history. Now a question spontaneously arises, why consider strategies without memory if in general those with memory are more expressive? The answer is simple: the computational complexity. There is also another factor that causes computational problems: the information that the agents have. We distinguish between two main classes of MAS: with perfect and imperfect information. In the former case, each agent has complete information about the system. Instead, we talk about imperfect information when there are some agents that do not have complete visibility over the system. Imperfect information is common in almost all MAS. Therefore, it seems likely that the most used scenario for MAS is in the context of imperfect information and memoryfull strategies. Unfortunately, in this setting the model checking problem for ATL and SL is undecidable in general [9]. Various techniques have been proposed to reduce complexity through symbolic or abstraction methods in which we have been personally involved [10, 11, 12, 13, 14, 15, 16]. In the context of imperfect information and memoryless strategies the model checking problem becomes tractable; in fact, for ATL and SL, it is PSPACE-COMPLETE [8, 17]. In the case of perfect information, model checking for ATL is polynomial [5], while for SL, with memoryfull strategies, it is non-elementary [7]. Due to the significance of this logic, several fragments have been proposed, such as Strategic Logic with Simple Goal [18] (co-authored by us), which shares the same model checking complexity as ATL but offers greater expressive power. Another intriguing research direction has introduced an extension of ATL with strategy contexts [19, 20]. Unlike the original semantics of ATL, in this logic, the strategy quantifiers do not reset previously selected strategies. Unfortunately, this gain in expressiveness comes at a significant cost, as the model checking problem is non-elementary, which is the same complexity class as model checking for SL. An alternative line of research that we have explored involves the introduction of a new class of strategies known as "natural strategies". The concept behind natural strategies is to embrace the perspective of bounded rationality and consider "simple" strategies when specifying agents' abilities. This concept has been introduced in both ATL and SL within the contexts of perfect [21, 22] and imperfect information [23, 24]. With this approach, in the worst-case scenario, the model checking complexity has been shown to be PSPACE. For more details on the model checking complexities, refer to the short survey we have developed [25].

Given this brief overview on multi-agent system verification, in the next section we will illustrate our main research results.

1.1 Our research work

As the title of the document suggests, we have worked on three main aspects in formal verification of MAS:

- how to model the system under exam;
- how to specify the properties of interest;
- how to verify that the model meets the specification, i.e. define a sound model checking procedure and study its complexity.

In what follows, we present our contributions for each field.

Formal Modeling. In this topic, our aim has been to provide formal mechanisms for describing specific settings. For example:

- In [26, 27, 28, 29], we have provided different tools to analyze cybersecurity problems in terms of multi-agent systems.
- In [30, 31, 32, 33, 34], we have developed some techniques to check whether there exists a backup strategy for an agent to achieve its objectives in the context of perfect and imperfect information.
- In [35, 36], we have defined a new concept of imperfect information. Instead of having partial visibility on the states of the MAS, we have provided a new notion of imperfect information over the actions.
- In [37, 38], we have equipped our model to share information between agents.
- In [39], we have introduced and solved concurrent multi-agent systems with parity objectives.
- In [21, 22], we have defined the concept of natural strategy, a strategy that fits the human's point a view. Then, in [23], we have studied natural strategies under imperfect information. Finally, we have applied natural strategies in concrete scenarios such as voting protocols [40, 41] and auctions [24].

In Chapter 2, we will detail some results obtained on natural strategies.

Formal Specification. In this context, our aim has been to define new logics with two orthogonal perspectives: gain expressiveness in terms of property specifications and/or decrease the model checking complexity. In particular, we have achieved the following results:

- In [42, 43, 44, 45], we have defined a graded version of SL to count how many strategies an agent, or a coalition of agents, has to achieve a strategic objective.
- In [18], we have defined a fragment of SL in which strategic operators, bindings operators, and temporal operators are coupled, called Strategy Logic with Simple-Goals. It has been shown that this fragment strictly subsume ATL and its model checking problem is PTIME-COMPLETE, as it is for ATL. Furthermore, in [46], we have provided a first analysis for implementing such a fragment.
- In [47, 48], we have defined an extension of Sabotage Logic in which we can erase a subset of edges.
- In [49], we have presented a logic that has the same expressive power of ATL but that is more succinct when verifying a strategic property without requiring knowledge of the exact coalitions involved in the specification.
- In [50], we have introduced Obstruction Logic (OL), a temporal logic to reason about models in which an agent (the Demon) can modify the structure of a system model by temporarily removing some edges that meet a quantitative threshold. Furthermore, in [51], we introduced OATL, an extension of ATL in which an agent can dynamically change the structure of a MAS.

In Chapter 3, we will detail the results obtained on obstruction logics.

Formal Verification. As mentioned before, model checking for MAS is undecidable in the context of imperfect information and memoryfull strategies. Given the relevance of this setting, even partial solutions to the problem can be useful. We have worked on this aspect in different directions:

- In [12, 52, 14], we have focused on an approximation on the visibility of the agents by defining a sound abstraction-refinement method.
- In [10, 11], we have defined a notion of bounded recall strategies and provided a preservation result to memoryfull strategies in three-valued semantics.
- In [16], we have studied the topological structure of the models and provided an approximation to temporal logics.
- In [53, 54, 15, 55], we have combined static and runtime verification techniques to determine decidability.

In addition, we have presented some other verification techniques, such as:

- In [56], we have presented an abstraction-refinement method to improve the model checking complexity for SL in practice.
- In [13], we have introduced a reduction of a fragment of ATL to first order logic. In this way, we have provided a technique to model check properties via SMT solvers.

- In [57, 58], we have proposed a reduction from epistemic dynamic logics to first order logic and provided some experimental results via SMT solvers.
- In [59], we have proposed a runtime solution in the imperfect information setting.
- In [60], we have provided a verification technique for models in which states are defined over databases.

In Chapter 4, we will detail some results aimed at establishing decidability within the context of imperfect information and memoryfull strategies.

Before delving into the details of some of our works, we conclude this chapter with some preliminary definitions that will be useful for reading the upcoming chapters.

Note that, given the imposed page limit, technical parts, such as proofs and algorithms, will not be presented in this document. For a more detailed description of the contents, we refer to the related articles published in journals and conferences (obviously cited in this document).

1.2 Preliminaries

In this section we introduce the standard semantics for the Alternating-time Temporal Logic ATL* and ATL [5]. To fix the notation, we assume sets $Ag = \{1, \ldots, m\}$ of *agents* and $AP = \{a_1, a_2, \ldots\}$ of *atomic propositions*, or simply atoms. Given a set U, $\overline{U}$ denotes its complement. We denote the length of a tuple v as $|v|$, and its i-th element as v_i. Let $last(v) = v_{|v|}$ be the last element in v. For $i \leq |v|$, let $v_{\geq i}$ be the suffix $v_i, \ldots, v_{|v|}$ of v starting at v_i and $v_{\leq i}$ its prefix $v_1, \ldots, v_i$. Notice that we start enumerations with index 1.

We begin by giving a formal account of multi-agent systems by means of concurrent game structures with imperfect information [5].

Definition 1.2.1. *Given sets Ag of agents and AP of atoms, a* concurrent game structure with imperfect information (iCGS) *is a tuple $M = \langle St, s_I, \{Act_i\}_{i \in Ag}, \{\sim_i\}_{i \in Ag}, \mathcal{P}, \delta, \mathcal{V} \rangle$ such that:*

- *St is a finite, non-empty set of* states, *with* initial state *$s_I \in St$.*
- *For every $i \in Ag$, Act_i is a finite, nonempty set of (individual)* actions.

  Let $Act = \bigcup_{i \in Ag} Act_i$ be the set of all actions, and $ACT = \prod_{i \in Ag} Act_i$ the set of all joint actions, *i.e., tuples of individual actions.*
- *For every $i \in Ag$, $\sim_i$ is a relation of* indistinguishability *between states, that is, an equivalence relation on St. Given states $s, s' \in St$, $s \sim_i s'$ iff s and s' are said to be* observationally indistinguishable *for agent i.*
- *The* protocol function *$\mathcal{P} : Ag \times St \to (2^{Act} \setminus \emptyset)$ defines the availability of actions so that for every $i \in Ag$, $s \in St$, (i) $\mathcal{P}(i, s) \subseteq Act_i$ and (i) $s \sim_i s'$ implies $\mathcal{P}(i, s) = \mathcal{P}(i, s')$.*
- *The* (deterministic) transition function *$\delta : St \times ACT \to St$ assigns a successor state $s' = \delta(s, \vec{\alpha})$ to each state $s \in St$, for every joint action $\vec{\alpha} \in ACT$ such that $\vec{\alpha}_i \in \mathcal{P}(i, s)$ for every $i \in Ag$, that is, $\vec{\alpha}$ is* enabled *at s.*
- *$\mathcal{L} : St \to 2^{AP}$ is a* labeling function.

By Definition 1.2.1 an iCGS describes the interactions of a group Ag of agents, starting from the initial state $s_I \in St$, according to the transition function δ. The latter is constrained by the availability of actions to agents, as specified by the protocol function $\mathcal{P}$. Furthermore, we assume that every agent i has imperfect information of the exact state of the system; so in any state s, i considers epistemically possible all states s' that are indistinguishable for i from s [6]. When every $\sim_i$ is the identity relation, *i.e.*, $s \sim_i s'$ iff $s = s'$, we obtain a standard CGS with perfect information [5].

Given a set $A \subseteq Ag$ of agents and a joint action $\vec{\alpha} \in ACT$, let $\vec{\alpha}_A$ (resp. $\vec{\alpha}_{\overline{A}}$) be the tuple comprising only of actions for the agents in A (resp. $\overline{A}$). We also write $\vec{\alpha}_i$ and $\vec{\alpha}_{\overline{i}}$ for $\vec{\alpha}_{\{i\}}$ and $\vec{\alpha}_{\overline{\{i\}}}$ respectively. Finally, for $\vec{\alpha}$ and $\vec{\beta}$ in ACT, $(\vec{\alpha}_A, \vec{\beta}_{\overline{A}})$ denotes the joint action where the actions for the agents in A (resp. $\overline{A}$) are taken from $\vec{\alpha}$ (resp. $\vec{\beta}$).

A history $h \in St^+$ is a finite (non-empty) sequence of states. The indistinguishability relations are extended to histories in a synchronous, pointwise way, *i.e.*, histories $h, h' \in St^+$ are *indistinguishable* for agent $i \in Ag$, or $h \sim_i h'$, iff (i) $|h| = |h'|$ and (ii) for all $j \leq |h|$, $h_j \sim_i h'_j$.

To reason about the strategic abilities of agents in iCGS, we use the Alternating-time Temporal Logic ATL* [5].

Definition 1.2.2. *The state (φ) and path (ψ) formulas in ATL* are defined as follows, where $a \in AP$ and $A \subseteq Ag$:*

$$\begin{array}{lcl} \varphi & ::= & a \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle\langle A \rangle\rangle \psi \\ \psi & ::= & \varphi \mid \neg\psi \mid \psi \wedge \psi \mid X\psi \mid (\psi U \psi) \end{array}$$

Formulas in ATL are all and only the state formulas.*

As customary, a formula $\langle\langle A \rangle\rangle \psi$ is read as "the agents in coalition A have a strategy to achieve ψ". The meaning of linear-time operators *next* X and *until* U is standard [2]. Operators *unavoidable* $[\![\Gamma]\!]$, *release* R, *eventually* F, and *globally* G can be introduced as usual.

The formulas in the ATL fragment of ATL* are obtained from Definition 1.2.2 by restricting path formulas ψ as follows, where φ is a state formula:

$$\psi \ ::= \ X\varphi \mid (\varphi U \varphi) \mid (\varphi R \varphi)$$

In the rest of the document we also consider the fragment of A-formulas, *i.e.*, formulas in which the strategic operator $\langle\langle A \rangle\rangle$ ranges only over some fixed coalition $A \subseteq Ag$. Furthermore, we also consider the existential and universal fragments.In particular, in the existential (resp. universal) fragment, formulas are only of the form $\langle\langle \Gamma \rangle\rangle \psi$ (resp. $[\![\Gamma]\!]\psi$) and the boolean negation operator is available only in front of atoms.

When giving a semantics to ATL* formulas we assume that agents are endowed with *uniform strategies* [61], *i.e.*, they perform the same action whenever they have the same information.

Definition 1.2.3. *A* uniform memoryfull strategy *for agent $i \in Ag$ is a function $\sigma_i : St^+ \to Act_i$ such that for all histories $h, h' \in St^+$, (i) $\sigma_i(h) \in \mathcal{P}(i, last(h))$; and (ii) $h \sim_i h'$ implies $\sigma_i(h) = \sigma_i(h')$.*

By Def. 1.2.3 any strategy for agent i has to return actions that are enabled for i. Also, whenever two histories are indistinguishable for i, then the same action is returned. Notice that, for the case of (perfect information) CGS, condition (ii) is satisfied by any function $\sigma_i : St^+ \to Act_i$.

Given an iCGS M, a *path* $\rho \in St^\omega$ is an infinite sequence $s_1 s_2 \ldots$ of states such that, for all $j \geq 1$, $s_{j+1} = \delta(s_j, \vec{\alpha})$ for some joint action $\vec{\alpha}$. Given a joint strategy $\sigma_A = \{\sigma_i \mid i \in A\}$, comprising of one strategy for each agent in coalition A, a path ρ is σ_A*-compatible* iff for every $j \geq 1$, $\rho_{j+1} = \delta(\rho_j, \vec{\alpha})$ for some joint action $\vec{\alpha}$ such that for every $i \in A$, $\vec{\alpha}_i = \sigma_i(\rho_{\leq j})$, and for every $i \in \overline{A}$, $\vec{\alpha}_i \in \mathcal{P}(i, \rho_j)$. Let $out(s, \sigma_A)$ be the set of all σ_A-compatible paths from state s.

We can now assign a meaning to ATL* formulas on iCGS.

Definition 1.2.4. *The satisfaction relation $\models$ for an iCGS M, state $s \in St$, path $\rho \in St^\omega$, atom $a \in AP$, state formula φ, and path formula ψ is defined as follows:*

$(M, s) \models a$	*iff*	$a \in \mathcal{L}(s)$
$(M, s) \models \neg\varphi$	*iff*	$(M, s) \not\models \varphi$
$(M, s) \models \varphi \wedge \varphi'$	*iff*	$(M, s) \models \varphi$ *and* $(M, s) \models \varphi'$
$(M, s) \models \langle\langle A \rangle\rangle \psi$	*iff*	*for some joint strategy* σ_A, *for all paths* $\rho \in out(s, \sigma_A)$, $(M, \rho) \models \psi$
$(M, \rho) \models \varphi$	*iff*	$(M, \rho_1) \models \varphi$
$(M, \rho) \models \neg\psi$	*iff*	$(M, \rho) \not\models \psi$
$(M, \rho) \models \psi \wedge \psi'$	*iff*	$(M, \rho) \models \psi$ *and* $(M, \rho) \models \psi'$
$(M, \rho) \models X\psi$	*iff*	$(M, \rho_{\geq 2}) \models \psi$
$(M, \rho) \models \psi U \psi'$	*iff*	*for some* $k \geq 1$, $(M, \rho_{\geq k}) \models \psi'$, *and for all* j, $1 \leq j < k$ *implies* $(M, \rho_{\geq j}) \models \psi$

We say that formula φ is *true* in an iCGS M, or $M \models \varphi$, iff $(M, s_I) \models \varphi$.

Notice that the satisfation clause for the release operator R can be derived as follows, by assuming that $\psi R \psi' ::= \neg(\neg\psi U \neg\psi')$:

$(M, \rho) \models \psi R \psi'$	iff	for all $k \geq 1$, $(M, \rho_{\geq k}) \models \psi'$, or for some $j \geq 1$, $(M, \rho_{\geq j}) \models \psi$, and for all j', $1 \leq j' \leq j$ implies $(M, p_{\geq j'}) \models \psi'$

Notice that the semantics discussed in this context aligns with the *objective interpretation* of ATL under imperfect information, as described in [61]. In contrast, the *subjective interpretation* requires a strategy to be successful for all states s' that are indistinguishable from the current state s. Both interpretations have been extensively examined in the model theory of logics for strategic reasoning, each presenting its own advantages and drawbacks. We refrain from delving into a comprehensive comparison of these approaches here and instead direct readers to [61] for further elaboration.

To conclude this chapter, we can state the model checking problem.

Definition 1.2.5 (Model Checking)**.** *Given an iCGS M and an ATL* formula ϕ, the model checking problem concerns determining whether $M \models \phi$.*

Chapter 2

Modeling: Natural Strategies

As outlined in the introduction, strategies within MAS are conceptualized as conditional plans and hold a central role in reasoning about purpose-driven agents. Formally, strategies are delineated as mappings from system histories to actions. While this approach holds mathematical validity and may suitably address the strategic capabilities of highly computational entities such as machines (robots, computer programs), we posit that it fails to accurately model human behavior. This discrepancy arises due to humans' inherent struggle with handling objects of combinatorial complexity. A human strategy ought to be relatively straightforward and intuitive, ensuring comprehension, memorization, and execution by the individual. This necessity is amplified when humans are required to devise strategies independently. Analogous concerns arise regarding the strategic capabilities of artificial agents constrained by limited memory and/or computational power, such as basic robots, sensors within autonomous sensor networks, and components of the Internet of Things. Consequently, we advocate for adopting "natural" abilities based on strategies with complexity within a set threshold.

Related Works. Works closely related to our proposal focus on modeling, specification, and reasoning about strategies of bounded agents. In this group, [62] investigates strategic properties of agents with bounded memory, while [63, 64, 65, 66] extend temporal and strategic logics to accommodate agents with bounded resources. Issues related to bounded rationality are also explored in [67, 68]. Papers examining the explicit representation of strategies are also relevant. This group is more extensive and includes extensions of ATL that explicitly reason about actions and strategies [69, 70], and logics combining features of temporal and dynamic logic [71]. A variant of STIT logic that enables reasoning about strategies and their performance within the object language is discussed in [72]. Furthermore, plans in agent-oriented programming can be seen as rule-based descriptions of strategies. Specifically, reasoning about agent programs using strategic logics has been investigated in [73, 74, 75, 76]. However, none of these works directly address the subject of our work: logic-based reasoning about agents' abilities in scenarios where natural representation and manageable complexity of strategies is crucial.

The rest of this chapter is structured as follows. First of all, as in [21, 22], in Section 2.1, we define the concept of natural strategy with and without memory in the context of perfect information. Then, as in [23], in Section 2.2, we introduce natural strategies in the context of imperfect information. Given these two representations, in Section 2.3, we introduce a variant of ATL that allows verifying if, for a coalition of agents, there exists a natural strategy with limited size to achieve a strategic objective. Finally, we provide results for the model checking problem of this logic.

2.1 Natural Strategies with perfect information

In this section, we focus on defining natural strategies with and without memory in the context of perfect information.

2.1.1 Memoryless

We start by defining the notion of *natural memoryless strategy* σ_i for agent i. The idea is to use a rule-based representation, with an ordered list of *condition-action* rules. The first rule whose condition holds in the current state is selected, and the corresponding action is executed. Formally, let $\mathcal{B}(\Sigma)$ be the set of Boolean formulas over alphabet Σ. We represent natural strategies by *ordered lists of guarded actions*, i.e., sequences of pairs (ϕ_j, α_j) such that:

1. $\phi_j \in \mathcal{B}(AP)$;
2. $\alpha_j \in \mathcal{P}(i, s)$ for every $s \in St$ such that $s \models \phi_j$.

That is, ϕ_j is a propositional condition on states of the CGS, and α_j is an action available to agent i in every state where ϕ_j holds. Moreover, we assume that the last pair in the list is $(\top, \alpha)$ for some $\alpha \in Act$, i.e., the last rule is guarded by a condition that will always be satisfied. Note that the action α must be available to agent i in every state of the system.

By $length(\sigma_i)$, we denote the number of guarded actions in σ_i. Moreover, $cond_j(\sigma_i)$ denotes the j-th guard (condition) on the list, and $act_j(\sigma_i)$ the corresponding action. Finally, $match(s, \sigma_i)$ is the smallest $j \leq length(\sigma_i)$ such that $s \models cond_j(\sigma_i)$ and $act_j(\sigma_i) \in \mathcal{P}(i, s)$. That is, $match(s, \sigma_i)$ matches state s with the first condition in σ_i that holds in s, and action available in s. Additionally, $dom(\phi) = \{p \in AP \mid p \in \phi\}$ stands for the set of atomic propositions that appear in condition ϕ, and $dom(\sigma_i) = \bigcup_{j=1,\ldots,length(\sigma_i)} dom(cond_j(\sigma_i))$ denotes the propositions occurring in σ_i. A *collective natural strategy* for a group of agents $A = \{1, \ldots, m\}$ is a tuple of individual natural strategies $\sigma_A = (\sigma_1, \ldots, \sigma_m)$. The "outcome" function $out(s, \sigma_A)$ returns the set of all paths that occur when agents in A execute strategy σ_A from state s onward. We emphasize that the outcome of σ_A collects *all* the paths consistent with σ_A. In particular, the opponents are not assumed to play a natural strategy; in fact, they are not assumed to play any strategy at all.

Example 2.1.1 (Ticket machine)**.** *The primary application domain we have in mind pertains to usability assessment. Consider, for example, a ticket vending machine situated at a railway station. Merely possessing a strategy to purchase the correct ticket is not sufficient. If the strategy proves overly intricate, the majority of users will struggle to navigate it effectively, rendering the machine practically ineffectual. To elucidate, consider the following specification presented as a natural strategy:*

1. $(\neg\text{ticket} \wedge \neg\text{selected} \wedge \neg\text{paid} \wedge \neg\text{error}, select)$;
2. $(\text{selected}, pay)$;
3. $(\top, idle)$.

It indicates that the customer selects a ticket only if no prior selection has been made (nor if the ticket has already been obtained or payment has been made). Following the selection, the customer proceeds with the payment process; otherwise, they remain idle.

By $compl(\sigma_i)$, we denote the complexity, or equivalently the size, of the strategy σ_i. Intuitively, the complexity of a strategy is understood as the level of sophistication of its representation. Several natural metrics can be used to measure the complexity of a strategy, given its representation from $(\mathcal{B}(AP) \times Act)^+$, e.g.:

- Number of used propositions: $compl_{\#}(\sigma_i) = |dom(\sigma_i)|$;
- Largest condition: $compl_{\max}(\sigma_i) = \max\{|\phi| \mid (\phi, \alpha) \in \sigma_i\}$;
- Total size of the representation: $compl_{\Sigma}(\sigma_i) = \sum_{(\phi,\alpha)\in\sigma_i} |\phi|$.

with $|\phi|$ being the number of symbols in ϕ, without parentheses.

From now on, we will focus on the last metric for complexity of strategies, which takes into account the total size of all the conditions used in the representation. That is, unless explicitly specified, we will assume $compl(\sigma_i) = compl_{\Sigma}(\sigma_i)$.

2.1.2 With Recall

Agents equipped with memory have the capability to make decisions based on the game's history, which encompasses the sequence of states experienced thus far. How can we effectively express conditions regarding such sequences? One approach is to utilize states within an automaton framework, as proposed in [77]. However, we advocate for a more intuitive representation for humans, achieved through regular expressions over propositional formulas.

Let $Reg(L)$ be the set of regular expressions over the language L (with the standard constructors $\cdot, \cup, *$ representing concatenation, nondeterministic choice, and finite iteration). A *natural strategy with recall* σ_i for agent i is a sequence of appropriate pairs from $Reg(\mathcal{B}(AP)) \times Act$. That is, it consists of pairs (r, α) where r is a regular expression over $\mathcal{B}(AP)$, and α is an action available in $last(h)$, i.e., $\alpha \in \mathcal{P}(i, last(h))$, for all histories $h \in St^+$ consistent with r.

Formally, given a regular expression r and the language $L(r)$ on words generated by r, a history $h = s_1 \dots s_n$ is consistent with r iff $\exists\, b \in L(r)$ such that $|h| = |b|$ and $\forall_{0 \le j \le n}\, h_j \models b_j$. Similarly to memoryless strategies, the last pair in the list is assumed to be simply $(\top^*, \alpha)$. Finally, $match(h, \sigma_i)$ is the smallest $k \le length(\sigma_i)$ such that $\forall_{0\le j\le|h|} h_j \models (cond_k(\sigma_i))_j$ and $act_k(\sigma_i) \in \mathcal{P}(i, h_j)$.

The metrics from memoryless strategies extend to strategies with recall and collective strategies with recall in the straightforward way. Additionally, we can define $compl_{\Sigma^*}(\cdot)$, a variant of the metric $compl_{\Sigma}(\cdot)$, that skips the initial $\top^*$ whenever it appears in a regular expression.

Example 2.1.2 (Wild West explorer). *Consider the following strategy with recall σ for a Wild West explorer:*

1. $(\mathsf{safe}^*, digGold)$;
2. $(\mathsf{safe}^* \cdot (\neg\mathsf{safe} \wedge \mathsf{haveGun}), shoot)$;
3. $(\mathsf{safe}^* \cdot (\neg\mathsf{safe} \wedge \neg\mathsf{haveGun}), run)$;

4. $(\top^* \cdot (\neg \mathsf{safe}) \cdot (\neg \mathsf{safe}), hide)$*;*

5. $(\top^*, idle)$*.*

Item (1) represents the guarded action in which safe *has held in all the states of the history. In such instances, the agent should proceed quietly to dig for gold. Alternatively, items (2) or (3) are used for each history in which* safe *held in all states except the last. In such scenarios, the agent should run away or shoot back depending on whether he has a gun. If it does not work (item (4)), the agent should hide. Conversely (item (5)), the agent should remain stationary and refrain from action. For the complexity, we have that* $compl_{\#}(\sigma) = 2$*,* $compl_{\max}(\sigma) = 8$*,* $compl_{\Sigma}(\sigma) = 27$*, and* $compl_{\Sigma^*}(\sigma) = 23$*.*

Before concluding this section, it is important to briefly discuss a relevant aspect of natural strategies in the perfect information context. In fact, given any CGS, these strategies introduce a sort of imperfect information on states where the set of atomic propositions is the same (see [22] for more details on this). However, it is important to emphasize that it is not possible to use the above-defined natural strategies in the imperfect information context and guarantee uniformity, particularly when two distinct states have distinct atomic propositions but are indistinguishable to an agent. For this reason, in the next section, we will address how to bridge this gap to define natural strategies in the context of imperfect information.

2.2 Natural Strategies with imperfect information

In this section, we show that the notion of *natural strategies*, introduced in [22], can be adapted to imperfect information scenarios in a very simple way.

2.2.1 Memoryless

We commence by introducing the concept of a *uniform natural memoryless strategy* σ_i for agent i. The essence lies in employing a rule-based framework, comprising an ordered list of *condition-action* rules. Upon evaluation, the first rule whose condition aligns with the present state is chosen, and the associated action is executed.

Formally, we define the set of *epistemic conditions* $\mathcal{E}$ for the agent i as follows:

$$\psi ::= \top \mid K_i\varphi$$
$$\varphi ::= a \mid \neg\varphi \mid \varphi \wedge \varphi \mid K_j\varphi$$

where a is an atomic proposition and j an agent.

So, we are talking about formulas that are prefixed by K_i and then possibly combined by Boolean operators. In other words, formulas ψ are always Boolean conditions on i's knowledge.

Given an iCGS M, a state $s \in St$, and an epistemic condition φ, we inductively define whether s satisfies φ ($s \models \varphi$) as follows:

- $s \models a$ iff $a \in \mathcal{L}(s)$;
- $s \models \neg\varphi$ iff $s \models \varphi$ does not hold;
- $s \models \varphi \wedge \varphi'$ iff $s \models \varphi$ and $s \models \varphi'$;
- $s \models K_i\varphi$ iff for all $s' \sim_i s$, it holds that $s' \models \varphi$.

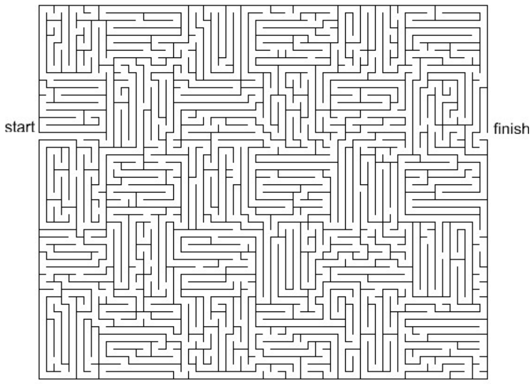

Figure 2.1: A maze with no loops.

We represent uniform natural strategies by *ordered lists of guarded actions*, i.e., sequences of pairs (ϕ_j, α_j) such that:

1. ϕ_j is an epistemic condition;
2. $\alpha_j \in \mathcal{P}(i, s)$ for every $s \in St$ such that $s \models \phi_j$.

That is, ϕ_j is an epistemic condition on states of the iCGS, and α_j is an action available to agent i in every state where ϕ_j holds. Moreover, as for the perfect information, we assume that the last pair on the list is $(\top, \alpha)$, with $\alpha \in \mathcal{P}(i, s)$, for all $s \in St$ and some $\alpha \in Act$.

It is easy to see that the strategies are uniform in the sense of [8], i.e., they specify the same actions in indistinguishable states.

Proposition 2.2.1 ([23]). *Given a uniform natural memoryless strategy σ_i and two states such that $s \sim_i s'$, we have that $act_{match(s,\sigma_i)}(\sigma_i) = act_{match(s',\sigma_i)}(\sigma_i)$.*

Proof. Take $\sigma_i = \big((\phi_1, \alpha_1), \ldots, (\phi_n, \alpha_n)\big)$ and any pair of states s, s' such that $s \sim_i s'$. Let $match(s, \sigma_i) = j$. That is, ϕ_j holds in s, but every ϕ_k for $k < j$ does not. Since all the formulas ϕ are either equal to $\top$ or begin with K_i, they must either hold in both s, s', or in none of them. Thus, we get that $match(s', \sigma_i) = j$, too. □

2.2.2 With Recall

A *uniform natural strategy with recall* σ_i for agent i is a sequence of appropriate pairs from $Reg(\mathcal{E}(AP)) \times Act$. That is, it consists of pairs (r, α) where r is a regular expression over $\mathcal{E}(AP)$, and α is an action available in $last(h)$, i.e. $\alpha \in \mathcal{P}(i, last(h))$, for all histories $h \in St^+$ consistent with r.

Again, we can observe that the strategies are uniform in the sense of [8], i.e., they specify the same actions in indistinguishable sequences of states.

Proposition 2.2.2 ([23]). *Given a uniform natural strategy with recall σ_i and two histories h, h' such that $|h| = |h'|$ and $\forall j \,.\, h_j \sim_i h'_j$, we have that $act_{match(h,\sigma_i)}(\sigma_i) = act_{match(h',\sigma_i)}(\sigma_i)$.*

As for the perfect information case, the metrics extend to strategies with recall and collective strategies with recall in the straightforward way.

Example 2.2.1 (Foggy maze). *Consider an agent rover rv, whose objective is to navigate through a maze, such as the one depicted in Figure 2.1. We assume the maze to be* perfect, *devoid of any loops. Furthermore, it is populated by several other hostile agents. At any given moment, each agent can opt to turn left (action $turn_L$), turn right ($turn_R$), move forward ($step$), or remain stationary ($wait$). Successful movement occurs if there are no obstacles or other agents blocking the path. Additionally, the rover has the ability to execute the $destroy$ action, eliminating any agent positioned directly in front of it, if present. Periodically, the maze may become enveloped in fog, during which agents experience complete blindness for 1 or 2 time units..*

Suppose an iCGS M representing the scenario in which states record the positions and orientations of all agents. In addition, two states are indistinguishable to an agent i if they coincide in i's position and orientation, and:

- *either both states are in foggy conditions,*
- *or both states are fog-free, and they agree on the content of the cell in front of i.*

The atomic propositions start *and* finish *mark the states where the rover is situated at the maze entry and exit, respectively. Propositions* wall *and* creature *identify states where the rover is facing a wall or another agent, respectively.*

The following natural strategy with recall guarantees that the rover gets through the maze (we use fog *as a shorthand for $\neg K_{rv}\mathsf{creature} \wedge \neg K_{rv}\neg\mathsf{creature}$ to simplify the notation):*

1. $(\top^* \cdot \mathsf{fog},\ wait)$;
2. $(\top^* \cdot K_{rv}\mathsf{creature},\ destroy)$;
3. $(\top^* \cdot K_{rv}\neg\mathsf{wall},\ step)$;
4. $(\top^* \cdot \neg K_{rv}\mathsf{wall} \cdot \mathsf{fog}^* \cdot K_{rv}\mathsf{wall},\ turn_L)$;
5. $(\top^* \cdot \neg K_{rv}\mathsf{wall} \cdot (\mathsf{fog}^* \cdot K_{rv}\mathsf{wall})^2,\ turn_R)$;
6. $(\top^* \cdot \neg K_{rv}\mathsf{wall} \cdot (\mathsf{fog}^* \cdot K_{rv}\mathsf{wall})^3,\ turn_R)$;
7. $(\top^*,\ turn_R)$.

That is, if fog is in the maze, the rover remains stationary until visibility improves. Upon encountering an adversary, it promptly eliminates it. In the event of confronting a wall, the rover initially attempts to turn left. If a wall persist in that direction, it iteratively turns right until discovering an unobstructed pathway.

The complexity of the strategy is $compl(\sigma_{rv}) = \{11+5+6+19+31+43+2\} = 117$. Note also that, if we add to the model an atomic proposition fog *that labels all the "foggy" states, we can use it instead of* fog *to specify the same behavior. This would reduce the complexity of the strategy to $\{4+5+6+12+17+22+2\} = 66$.*

Finally, we observe that a classic strategy may result in a very ineffective traversal of the maze, i.e., the number of steps between the start and the exit can be large. Still, the natural strategy above has two important advantages. First, it is much simpler – and therefore much easier to store and use – than the combinatorial strategy that specifies the right choice for every position of the rover. Secondly, it is general in the sense that it does not depend on the actual shape of the labyrinth.

2.3 Natural Strategies in ATL

Natural ATL (NatATL, for short) is obtained by replacing in ATL the modality $\langle\langle A\rangle\rangle$ with the bounded strategic modality $\langle\langle A\rangle\rangle^{\leq k}$. Intuitively, $\langle\langle A\rangle\rangle^{\leq k}\phi$ reads as "coalition A has a collective strategy of *size less or equal than* k to enforce the property ϕ." As in ATL, the formulas of NatATL make use of classical temporal operators: X ("in the next state"), G ("always from now on"), F ("now or sometime in the future"), U (strong "until"), and W (weak "until").

Thus, the language of NatATL can be defined by the following grammar:

$$\varphi ::= a \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle\langle A\rangle\rangle^{\leq k} X\varphi \mid \langle\langle A\rangle\rangle^{\leq k}\varphi U\varphi \mid \langle\langle A\rangle\rangle^{\leq k}\varphi W\varphi$$

where $A \subseteq Ag$, $k \in \mathbb{N}$, and $a \in AP$.

Given an iCGS M, a state $s \in St$, a path ρ, atom $a \in AP$, and $k \in \mathbb{N}$, the semantics of NatATL is defined as follows:

$(M, s) \models a$	iff	$a \in \mathcal{L}(s)$
$(M, s) \models \neg\varphi$	iff	$(M, s) \not\models \varphi$
$(M, s) \models \varphi \wedge \varphi'$	iff	$(M, s) \models \varphi$ and $(M, s) \models \varphi'$
$(M, s) \models \langle\langle A\rangle\rangle^{\leq k} X\varphi$	iff	for some joint natural strategy σ_A, such that $compl(\sigma_A) \leq k$ and, for all path $\rho \in out(s, \sigma_A)$, $(M, \rho_2) \models \varphi$
$(M, s) \models \langle\langle A\rangle\rangle^{\leq k}\varphi U\varphi'$	iff	for some joint natural strategy σ_A, such that $compl(\sigma_A) \leq k$ and, for all path $\rho \in out(s, \sigma_A)$, $(M, \rho_i) \models \varphi'$ for some $i \geq 1$ and $(M, \rho_j) \models \varphi$ for all $1 \leq j < i$
$(M, s) \models \langle\langle A\rangle\rangle^{\leq k}\varphi W\varphi'$	iff	for some joint natural strategy σ_A, such that $compl(\sigma_A) \leq k$ and, for all path $\rho \in out(s, \sigma_A)$, either $(M, \rho_i) \models \varphi'$ for some $i \geq 1$ and $(M, \rho_j) \models \varphi$ for all $1 \leq j < i$ or $(M, \rho_i) \models \varphi$ for all $i \geq 1$

Once again, we emphasize that when evaluating the formula $\langle\langle A\rangle\rangle^{\leq k}\phi$, we do not assume the opponents to play a natural strategy (bounded or otherwise). This corresponds to the pessimistic approach to evaluating ability based on 'surely winning': the agents in A win only if they have a strategy that wins against every — even accidental — behavior of the rest of the system.

2.3.1 Model Checking Results

We begin by showing the model checking of NatATL under the assumption that the complexity bounds k used in formulas are constant or bounded. In other words, they are not a parameter of the model checking problem. Under this restriction, one can show a polynomial reduction to the model checking problem for CTL formulas. In consequence, we obtain the following result.

Theorem 2.3.1 ([22, 23]). *The model checking problem for NatATL with memoryless natural strategies and fixed k is in* PTIME *with respect to the size of the model and the length of the formula.*

For the general case, in which k is a variable of the problem, we need of an oracle for each strategic operator involved in the formula. Thus, we can define a polynomial algorithm that calls a non-deterministic algorithm that generates a natural strategy. In consequence, we obtain the following result.

Theorem 2.3.2 ([22, 23]). *The model checking problem for NatATL with memoryless natural strategies is Δ_2^P-complete with respect to the size of the model, the length of the formula, and the maximal bound k in the formula.*

In the context of natural strategies with recall, when the bound of the strategies is fixed or bounded by a constant, we can still use an oracle. This leads to the following result.

Theorem 2.3.3 ([22, 23]). *The model checking problem for NatATL with natural strategies with recall and fixed k is in Δ_2^P with respect to the size of the model and the length of the formula.*

Applying the same algorithm as proposed for the fixed k case to the general scenario with a variable k would result in an exponential algorithm. Consequently, when analyzing the memory space required to solve the algorithm, we arrive at the following outcome.

Theorem 2.3.4 ([22, 23]). *The model checking problem for NatATL with natural strategies with recall is in PSPACE with respect to the size of the model, the length of the formula, and the maximal bound k in the formula.*

Notice that, the above results hold for both perfect and imperfect information contexts.

Before concluding this chapter, we would like to clarify that unfortunately, our logic cannot use the ATL model checking algorithm, which is in PTIME. This is because, with the ATL fixed-point algorithm, we cannot determine the complexity of a strategy, which is the key notion of the NatATL strategic operator. However, as mentioned at the beginning of this chapter, the importance of introducing NatATL lies in the fact that in certain contexts where there is a need to produce a compact strategy, using ATL model checking may not be sufficient. In fact, verifying that a strategy exists but is not usable only gives us a false positive in the verification process.

Chapter 3

Specification: Logics in Cybersecurity

The logics outlined in this chapter have direct applications in the cybersecurity domain, particularly in the design of active security response strategies during ongoing attacks. Our models facilitate the encapsulation of interactions between attackers, whose potential actions are modeled using *Attack Graphs* [78], and defenders that can dynamically deploy *Moving Target Defense (MTD)* mechanisms [79] based on these attack graphs.

Related Works. In recent years, some works have focused on the strategic abilities of agents in dynamic game models. For instance, [80] addresses planning in a dynamic, but predictable, environment. However, it does not allow agents to select specific subsets of successors, unlike our approach. Additionally, [80] permanently removes edges, while our method only temporarily deactivates them based on quantitative information. Sabotage games and Sabotage Modal Logic [81, 82, 83] study the computational complexity of graph-reachability problems where agents can erase edges. Our approach is incomparable with Sabotage games since we give the ability to temporarily select subsets of edges while in Sabotage games the saboteur can erase only one edge at each turn. In [84], NTL, a temporal logic for normative systems, is introduced. NTL evaluates CTL formulas on models with deleted arcs, but the assignment function is non-local and non-quantitative. Module checking [85, 86, 87] is also related to our logic. Although there are similarities with our contribution, the approaches are orthogonal. In our logics, each state is controlled by the environment (the Demon), and we seek a winning strategy for the environment, not whether all strategies are winning.

The rest of this chapter is structured as follows. In Section 3.1, we introduce Obstruction Logic (OL), an extension of CTL that allows an agent (the Demon) to modify the structure of a model in a two-agent setting. Meanwhile, in Section 3.2, we present an extension of ATL, called Obstruction Alternating-time Temporal Logic (OATL), which allows an agent to modify the structure of a model in a multi-agent context. For both logics, we demonstrate that the model checking complexity is polynomial, thereby showing that we increase expressiveness with respect to CTL and ATL without incurring any additional computational costs.

3.1 Obstruction Logic

First of all, we present the syntax of our logic.

Definition 3.1.1. *Formulas of Obstruction Logic (OL, for short) are defined by the following* *grammar:*

$$\varphi ::= a \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle\langle \ddagger_n \rangle\rangle X\varphi \mid \langle\langle \ddagger_n \rangle\rangle(\varphi U \varphi) \mid \langle\langle \ddagger_n \rangle\rangle(\varphi R \varphi)$$

where $a \in AP$ and $n \in \mathbb{N}$.

The boolean and temporal connectives can be derived as usual. The intuitive meaning of a formula $\langle\langle \ddagger \rangle\rangle\varphi$ is "there is a demonic strategy such that all paths of the model that are compatible with the strategy satisfy φ" where "demonic strategy" means a strategy for disabling arcs. Formulas of OL will be interpreted over obstruction models. The definition follows.

Definition 3.1.2. *An Obstruction Model (OM) is a pair $M_\ddagger = \langle M, \$\rangle$, where M is a CGS and $\$: St \times ACT \to \mathbb{N}^+$ is the (partial) cost function. Such function associates to any state s and joint action $\vec{\alpha}$ such that $\delta(s, \vec{\alpha})$ is defined, a positive natural number $\$(s, \vec{\alpha})$.*

Our logic is designed to encompass strategies for obstruction models in which one of the two agents, referred to as the Demon, possesses the ability to temporarily deactivate arcs within the model. Specifically, given a history h, a demonic strategy selects a subset of arcs adjacent to $last(h)$, ensuring that the sum of their weights does not exceed a predefined threshold. The arcs chosen by the demonic strategies are then temporarily removed from the set of arcs available for selection by the other agent. In this manner, the structure of the OM is altered by the actions of the Demon. We formally define the notion of a demonic strategy as follows.

Definition 3.1.3. *Let $n \in \mathbb{N}$ be a natural number, a* demonic n-strategy *is a function $\sigma_\ddagger : St^+ \to 2^{ACT}$ that given an history h, returns a subset of joint actions $AC \in ACT$ such that:*

1. *for each $\vec{\alpha} \in AC$, for each $i \in Ag$, $\vec{\alpha}_i \in \mathcal{P}(i, last(h))$;*
2. *$(\sum_{\vec{\alpha} \in AC} \$(last(h), \vec{\alpha})) \leq n$.*

As it happens for the logic ATL, the notion of path that is compatible with a strategy, is the central pivot of the semantic of OL formulas. We define this notion by saying that: a path ρ is compatible with a n-strategy $\sigma_\ddagger$ if for all $i \geq 1$ we have that $\delta(\rho_i, \vec{\alpha}) = \rho_{i+1}$ implies $\vec{\alpha} \notin \sigma_\ddagger(\rho_{\leq i})$. Given a state s and a n-strategy $\sigma_\ddagger$, $Out(s, \sigma_\ddagger)$ denotes the set of paths whose first state is s and that are compatible with $\sigma_\ddagger$. We can now define the semantics of OL formulas.

Definition 3.1.4. *The satisfaction relation between a model $M_\ddagger$, a state s of $M_\ddagger$, and a formula φ is defined by induction on the structure of φ:*

$(M_\ddagger, s) \models a$	*iff*	$a \in \mathcal{L}(s)$
$(M_\ddagger, s) \models \neg\varphi$	*iff*	$(M_\ddagger, s) \not\models \varphi$
$(M_\ddagger, s) \models \varphi \wedge \varphi'$	*iff*	$(M_\ddagger, s) \models \varphi$ *and* $(M_\ddagger, s) \models \varphi'$
$(M_\ddagger, s) \models \langle\langle \ddagger_n \rangle\rangle X\varphi$	*iff*	*for some n-strategy* $\sigma_\ddagger$, *for all* $\rho \in Out(s, \sigma_\ddagger)$, $(M_\ddagger, \rho_2) \models \varphi$
$(M_\ddagger, s) \models \langle\langle \ddagger_n \rangle\rangle(\varphi U \varphi')$	*iff*	*for some n-strategy* $\sigma_\ddagger$, *for all* $\rho \in Out(s, \sigma_\ddagger)$ *there is a* $j \geq 1$, $(M_\ddagger, \rho_j) \models \varphi'$ *and for all* $1 \leq k < j$, $(M_\ddagger, \rho_k) \models \varphi$
$(M_\ddagger, s) \models \langle\langle \ddagger_n \rangle\rangle(\varphi R \varphi')$	*iff*	*for some n-strategy* $\sigma_\ddagger$, *for all* $\rho \in Out(s, \sigma_\ddagger)$ *either* $(M_\ddagger, \rho_i) \models \varphi'$ *for all* $i \geq 1$ *or there is a* $k \geq 1$, $(M_\ddagger, \rho_k) \models \varphi$ *and* $(M_\ddagger, \rho_j) \models \varphi'$ *for all* $1 \leq j \leq k$

In [50], we demonstrate that the fixed-point characterization for temporal operators also holds in OL, and consequently, we can produce an algorithm that is an extension of CTL, yielding the following result.

Theorem 3.1.1 ([50]). *The model checking problem for OL is* PTIME-COMPLETE.

3.2 Obstruction Alternating-time Temporal Logic

First, we now introduce the syntax of this new logic.

Definition 3.2.1. *State φ and path ψ formulas of Obstruction Alternating-time Temporal Logic (OATL, for short) are defined by the following grammar:*

$$\varphi ::= a \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle\langle A_n^{\ddagger} \rangle\rangle \psi$$
$$\psi ::= X\varphi \mid \varphi U \varphi \mid \varphi R \varphi$$

where $a \in AP$, A is any subset of Ag, and $n \in \mathbb{N}$.

Formulas of OATL are all and only the state formulas.

As for OL, an obstruction model is a CGS provided with a function that assigns a cost (a positive natural number) to any pair composed of a state and a joint action that defines a transition from the given state. Thus, a model is a labeled, directed weighed graph. Now, we have all the ingredients to provide the semantics of OATL.

Definition 3.2.2. *The satisfaction relation $(M_{\ddagger}, s) \models \varphi$ between an obstruction model $M_{\ddagger}$, a state s of $M_{\ddagger}$, and a state formula φ is defined as follows:*

$(M_{\ddagger}, s) \models a$	*iff*	$a \in \mathcal{L}(s)$
$(M_{\ddagger}, s) \models \neg\varphi$	*iff*	$(M_{\ddagger}, s) \not\models \varphi$
$(M_{\ddagger}, s) \models \varphi \wedge \varphi'$	*iff*	$(M_{\ddagger}, s) \models \varphi$ *and* $(M_{\ddagger}, s) \models \varphi'$
$(M_{\ddagger}, s) \models \langle\langle A_n^{\ddagger} \rangle\rangle \psi$	*iff*	*for some demonic n-strategy* $\sigma_{\ddagger}$*, for all strategies* σ_A *if* $Out(s, \sigma_{\ddagger}) \cap out(s, \sigma_A) \neq \emptyset$*, then there is a* $\rho \in Out(s, \sigma_{\ddagger}) \cap out(s, \sigma_A)$*,* $(M_{\ddagger}, \rho) \models \psi$

The satisfaction relation $(M_{\ddagger}, \rho) \models \varphi$ between a model $M_{\ddagger}$, a path ρ of $M_{\ddagger}$, and path formula ψ is defined as follows:

$(M_{\ddagger}, \rho) \models X\varphi$	*iff*	$(M_{\ddagger}, \rho_2) \models \varphi$
$(M_{\ddagger}, \rho) \models \varphi U \varphi'$	*iff*	*there is an* $i \geq 1$*,* $(M_{\ddagger}, \rho_i) \models \varphi'$ *and* $(M_{\ddagger}, \rho_j) \models \varphi$ *for all* $1 \leq j < i$
$(M_{\ddagger}, \rho) \models \varphi R \varphi'$	*iff*	*either* $(M_{\ddagger}, \rho_i) \models \varphi'$ *for all* $i \geq 1$ *or there is a* $k \geq 1$*,* $(M_{\ddagger}, \rho_k) \models \varphi$ *and* $(M_{\ddagger}, \rho_j) \models \varphi'$ *for all* $1 \leq j \leq k$

The idea behind the strategic operator is to existentially quantify over the Demon's strategy, universally over the strategies of the other agents (who, from the Demon's perspective, are adversaries), and finally to existentially quantify over the paths. Note that the last two quantifications are semantically equivalent to the universal operator of ATL. We made this choice to be able to verify if there exists a strategy for the Demon against all those of the other agents. Furthermore, we remark that since any demonic n-strategy can select only a strict subset of the set of joint actions available at $last(h)$ for a given history h, we can never have that $Out(\sigma_{\ddagger}, s) \cap out(\sigma_A, s) = \emptyset$ for every strategy σ_A.

As shown for OL, we can also find the fixed-point characterization for OATL, and thus demonstrate the following result.

Theorem 3.2.1 ([51]). *The model-checking problem for OATL is* PTIME-COMPLETE.

3.3 Use Case Scenario in Cybersecurity

To conclude this chapter, we show a toy example[1] of how obstruction logics are useful for defining properties in the context of cybersecurity.

In a wireless network scenario involving three users (referred to as Alice, Bob, and David), each user has the capability to alter their status within the network, including their position and granted privileges. These modifications can be accomplished either through legitimate requests to the network or through malicious attacks. Among the users, two (Alice and Bob, forming coalition C) are malicious and aim to compromise the network's integrity. Their objective is to orchestrate a situation where Alice obtains root access on a specific network server, Bob gains root access on another server, and David successfully requests and obtains a specific resource from the network. At each moment, Alice and Bob have the option to either remain inactive ($\star$) or execute an attack on the network to obtain root privileges on their desired server (att). David, on the other hand, can choose to either make a specific request (req_i) to the network or take no action ($\star$). Let r_a be the atomic proposition expressing that Alice is root of the needed server, r_b be the atomic proposition expressing that Bob is root of the needed server, and that g_1 and g_2 be the atomic proposition expressing the fact that David has been granted access to resources 1 and 2, respectively. Given these premises, a possible interaction between Alice, Bob, and David is depicted in the model $M_\ddagger$ of Figure 3.1.

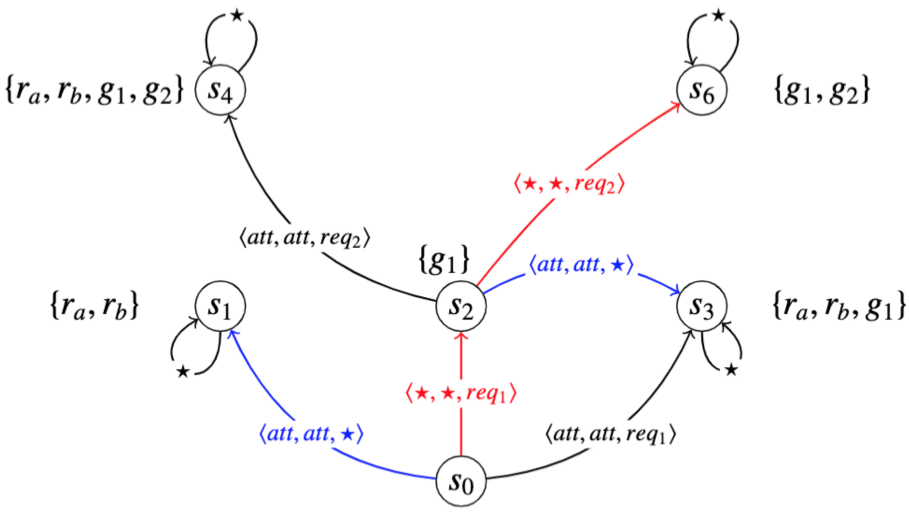

Figure 3.1: A model $M_\ddagger$ depicting the considered attack scenario. The symbol $\star$ stands for $\{\star, \star, \star\}$.

In the given scenario, let's suppose there is an intelligent agent, referred to as the defender, capable of temporarily blocking the collective actions of users. Blocking these actions incurs specific costs, which may vary depending on the nature of each user's action. For instance, considering Figure 3.1, joint actions corresponding to blue edges have a cost of 1, black edges have a cost of 2, and red edges have a cost of 3. Given an initial state s_0 in the model, the question is whether there exists a strategy for the defender such that, for any strategy adopted by coalition C (Alice and Bob), there is at least one scenario where Alice and Bob are never in

[1]Examples of such real case studies, at least for the attack part, can be found in [88].

a position to launch the fatal attack. By using OATL, we can express this property as follows:

$$\varphi = \langle\langle C_n^{\ddagger} \rangle\rangle (\bot R \neg (r_a \wedge r_b \wedge g_1))$$

Clearly, with φ we can determine whether there exists a winning strategy for the coalition C to compromise the network. In particular, if the formula holds, it means that coalition C does not have a winning strategy, whereas if φ turns out to be false, then coalition C has a winning strategy to attack the network regardless of any countermeasure the defender may take. On our example, the truth value of the formula φ is determined by the value n. In particular, the minimum n for which $\langle\langle C_n^{\ddagger} \rangle\rangle (\bot R \neg (r_a \wedge r_b \wedge g_1))$ holds is 3. In fact, the 3-demonic strategy $\sigma_{\ddagger}$ selecting the joint action $\vec{\alpha}$ such that $\vec{\alpha}_{Alice} = \star$, $\vec{\alpha}_{Bob} = \star$ and $\vec{\alpha}_{David} = req_1$ given any history h where $last(h) = s_0$, is a good candidate. Indeed, the set of paths in $Out(s_0, \sigma_{\ddagger})$ contains $s_0 \cdot s_1^{\omega}$ and $s_0 \cdot s_3^{\omega}$, and the strategies σ_C such that $Out(s_0, \sigma_{\ddagger}) \cap Out(s_0, \sigma_C) \neq \emptyset$ are those in which Alice and Bob both choose att on s_0. For all these strategies, there is a path (i.e., $s_0 \cdot s_1^{\omega}$) that satisfies $\bot R \neg (r_a \wedge r_b \wedge g_1)$.

Chapter 4

Verification: Decidable Fragments

As mentioned in the introduction, the model checking problem of ATL with imperfect information and memoryfull strategies is undecidable in general. In this chapter, we will present sound but incomplete approximation methods that allow us to make fragments of this problem decidable. In particular, in Section 4.1, we present an abstraction-refinement method related to agent information. In Section 4.2, we focus on the other problem that makes model checking for ATL undecidable, namely the memory of strategies. In particular, we show a method to approximate perfect recall strategies through bounded recall strategies. Finally, in Section 4.3, we present another approach in which model abstraction and formula approximation are used to find the decidability.
Related Works. Several approaches for verifying specifications in ATL under imperfect information and perfect recall have been proposed recently. One line restricts how information is shared among agents to retain decidability [89]. Another line limits interactions to public actions only [90, 91]. These approaches differ from ours as they seek decidable classes, whereas we define a general verification procedure for the whole class of iCGS. An abstraction-refinement framework for CTL over three-valued semantics was studied in [92, 93] and hierarchical systems are considered in [94]. [95] introduces an abstraction-refinement technique for full μ-calculus. [96] and [97] present abstraction-refinement for network games with perfect information and two-player games, respectively. [98] studies games with incomplete information for safety goals, using abstraction and refinement to transform imperfect information games into perfect information ones. Model checking MAS by abstraction in an epistemic context is discussed in [99, 100]. Three-valued abstractions for verifying ATL properties are found in [101, 102, 20, 103]. These methods focus on decidable settings, interpreting ATL under perfect information [101, 92] or considering non-uniform strategies [102, 20, 103], aiming to speed up verification tasks. Finally, in [104], a multi-valued semantics for ATL* is presented as a conservative extension of the classical two-valued variant, addressing the model checking problem for perfect information games and providing results for imperfect information games.

4.1 Approximate the information

At the core of our contribution is the notion that, within a three-valued semantics framework, MAS with imperfect information can be effectively approximated by perfect information systems. This abstraction process empowers us to devise a robust yet incomplete verification procedure for the strategy logics ATL and ATL*, operating under conditions of imperfect information and perfect recall. In essence, given an iCGS representing a MAS, we construct a perfect information abstraction that preserves satisfaction for a three-valued variant of ATL*. As we will show, if the ATL* specification holds true (or false) in the perfect information abstraction, it similarly holds true (or false) in the original iCGS. Conversely, if the specification remains undefined, we have the opportunity to refine the abstraction in pursuit of assigning it a defined truth value. It is important to acknowledge that the original model checking problem is undecidable. Therefore, there is no guarantee that by successive refinements, the truth or falsity of a given property can ever be fully established in a general context. However, the procedure we have outlined offers a constructive method for partially model checking ATL* under conditions of imperfect information and perfect recall. This approach enables us to make progress towards verifying properties of MAS, even though complete verification may not always be attainable.

4.1.1 Three-valued semantics

Here, we introduce a generalization of iCGS in terms of over- and under-approximations. Then, we develop a three-valued semantics for ATL*, and show that it conservatively extends the two-valued semantics presented in the preliminaries. In the rest of the section, for $x = may$ (resp. $must$), we set $\overline{x} = must$ (resp. may).

Definition 4.1.1. *Given sets Ag of agents and AP of atoms, a* generalized iCGS *is a tuple $M = \langle St, s_I, \{Act_i\}_{i \in Ag}, \{\sim_i\}_{i \in Ag}, \mathcal{P}^{may}, \mathcal{P}^{must}, \delta^{may}, \delta^{must}, \mathcal{L} \rangle$ such that:*

1. *St, s_I, $\{Act_i\}_{i \in Ag}$, $\{\sim_i\}_{i \in Ag}$ are defined as in Definition 1.2.1.*
2. *$\mathcal{P}^{may}$ and $\mathcal{P}^{must}$ are* protocol functions *from $Ag \times St$ to $2^{Act} \setminus \emptyset$ such that for every $i \in Ag$ and $s \in St$, (i) $\mathcal{P}^{must}(i, s) \subseteq \mathcal{P}^{may}(i, s) \subseteq Act_i$ and (ii) $s \sim_i s'$ implies $\mathcal{P}^x(i, s) = \mathcal{P}^x(i, s')$.*
3. *δ^{may} and δ^{must} are* transition relations *on $St \times ACT \times St$ such that $s' \in \delta^x(s, \vec{\alpha})$ is defined for some $s' \in St$ only if $\vec{\alpha}_i \in \mathcal{P}^x(i, s)$ for every $i \in Ag$. Moreover, $\delta^{must}(s, \vec{\alpha}) \subseteq \delta^{may}(s, \vec{\alpha})$.*
4. *$\mathcal{L} : St \times AP \to \{\top, \bot, \mathrm{uu}\}$ is a* three-valued labelling function.

Intuitively, $must$-components are more stringent than may-components: $must$-transitions can be seen as under-approximations of the actual transitions in the iCGS, while may-transitions can be considered as over-approximations. The undefined value uu can be interpreted in various ways, for instance, unknown, unspecified, or inconsistent, depending on the application at hand. We say that the truth value τ is *defined* whenever $\tau \neq \mathrm{uu}$. In the case that under- and over-approximations coincide, *i.e.*, $\mathcal{P}^{may} = \mathcal{P}^{must}$ and $\delta^{may} = \delta^{must}$ are functions, and the truth value of every atom is defined, then we have a standard iCGS as per Definition 1.2.1. On the other hand, if each equivalence relation $\sim_i$ is the identity, then we have a generalized CGS.

Now, we introduce $must$- and may-strategies.

Definition 4.1.2. *For $x \in \{may, must\}$, a* uniform x-strategy with perfect recall *for agent $i \in Ag$ is a function $\sigma_i^x : St^+ \to Act_i$ such that for every history $h, h' \in St^+$, (i) $\sigma_i^x(h) \in \mathcal{P}^x(i, last(h))$; and (ii) $h \sim_i h'$ implies $\sigma_i^x(h) = \sigma_i^x(h')$.*

Here we distinguish between may and $must$ strategies to over- and under-approximate the strategic abilities of agents. Again, the distinction collapses in the case of standard (two-valued) iCGS.

For $x \in \{may, must\}$ and a joint strategy $\sigma_A^x = \{\sigma_i^x \mid i \in A\}$, a path $\rho \in St^\omega$ is σ_A^x-*compatible* iff for every $j \geq 1$, $\rho_{j+1} = \delta^{\overline{x}}(\rho_j, \vec{\alpha})$ for some joint action $\vec{\alpha}$ such that for every $i \in A$, $\vec{\alpha}_i = \sigma_i^x(\rho_{\leq j})$, and for every $i \notin A$, $\vec{\alpha}_i \in \mathcal{P}^{\overline{x}}(i, \rho_j)$. Then, $out(s, \sigma_A^x)$ is the set of all σ_A^x-compatible paths starting from s. Intuitively, when computing the outcomes of a joint strategy σ_A^{must} from state s, we take a "conservative" approach regarding the abilities of agents in A. This involves considering only actions enabled according to the under-approximated protocol $\mathcal{P}^{must}$, while maintaining an "optimistic" stance about the capabilities of agents in $\overline{A}$, as provided by the over-approximated protocol $\mathcal{P}^{may}$ and transition δ^{may} functions. Conversely, for $out(s, \sigma_A^{may})$, the reasoning is reversed. However, since σ_A^{may} returns may-actions and paths in $out(s, \sigma_A^{may})$ are generated by considering the δ^{must}-transitions, $out(s, \sigma_A^{may})$ may turn out to be empty. In such cases, according to the definition of the semantics below, the formula will be undefined.

Formally we define the three-valued semantics for ATL* as follows.

Definition 4.1.3. *The three-valued satisfaction relation $\models_3$ for an iCGS M, state $s \in St$, path $\rho \in St^\omega$, atom $a \in AP$, $\tau \in \{\top, \bot\}$, state formula φ, and path formula ψ is defined as follows:*

$((M, s) \models_3 a) = \tau$	*iff*	$\mathcal{L}(s, a) = \tau$
$((M, s) \models_3 \neg\varphi) = \tau$	*iff*	$((M, s) \models_3 \varphi) = \neg\tau$
$((M, s) \models_3 \varphi \wedge \varphi') = \top$	*iff*	$((M, s) \models_3 \varphi) = \top$ *and* $((M, s) \models_3 \varphi') = \top$
$((M, s) \models_3 \varphi \wedge \varphi') = \bot$	*iff*	$((M, s) \models_3 \varphi) = \bot$ *or* $((M, s) \models_3 \varphi') = \bot$
$((M, s) \models_3 \langle\langle A\rangle\rangle\psi) = \top$	*iff*	*for some joint strategy* σ_A^{must}, *for all paths* $\rho \in out(s, \sigma_A^{must})$, $((M, \rho) \models_3 \psi) = \top$
$((M, s) \models_3 \langle\langle A\rangle\rangle\psi) = \bot$	*iff*	*for every joint strategy* σ_A^{may}, *for some path* $\rho \in out(s, \sigma_A^{may})$, $((M, \rho) \models_3 \psi) = \bot$
$((M, \rho) \models_3 \varphi) = \tau$	*iff*	$((M, \rho_1) \models_3 \varphi) = \tau$
$((M, \rho) \models_3 \neg\psi) = \tau$	*iff*	$((M, \rho) \models_3 \psi) = \neg\tau$
$((M, \rho) \models_3 \psi \wedge \psi') = \top$	*iff*	$((M, \rho) \models_3 \psi) = \top$ *and* $((M, \rho) \models_3 \psi') = \top$
$((M, \rho) \models_3 \psi \wedge \psi') = \bot$	*iff*	$((M, \rho) \models_3 \psi) = \bot$ *or* $((M, \rho) \models_3 \psi') = \bot$
$((M, \rho) \models_3 X\psi) = \tau$	*iff*	$((M, \rho_{\geq 2}) \models_3 \psi) = \tau$
$((M, \rho) \models_3 \psi U \psi') = \top$	*iff*	*for some* $k \geq 1$, $((M, \rho_{\geq k}) \models_3 \psi') = \top$, *and for all* j, $1 \leq j < k$ *implies* $((M, p_{\geq j}) \models_3 \psi) = \top$
$((M, p) \models_3 \psi U \psi') = \bot$	*iff*	*for all* $k \geq 1$, $((M, p_{\geq k}) \models_3 \psi') = \bot$, *or for some* $j \geq 1$, $((M, p_{\geq j}) \models_3 \psi) = \bot$, *and for all* j', $1 \leq j' \leq j$ *implies* $((M, p_{\geq j'}) \models_3 \psi') = \bot$

In all other cases the value of ϕ is uu.

Observe that, in the clauses for ATL* operators $must$-strategies are used to check the truth of formulas, while may-strategies appear in the clauses for falsehood. Specifically, to check whether $((M, s) \models_3 \langle\langle A\rangle\rangle\psi) = \top$ we consider all paths in $out(s, \sigma_A^{must})$, which are defined by δ^{may}-transitions. This restricts the choices available to coalition A, while increasing the number of paths in which the formula needs to be satisfied. Similarly, to verify whether $((M, s) \models_3 \langle\langle A\rangle\rangle\psi) = \bot$ we need to use δ^{must}-transitions over the paths in $out(s, \sigma_A^{may})$, so as to decrease the number of candidates witnessing the falsehood of the formula. Notice also that, as regards Boolean operators, our semantics correspond to Kleene's three-valued logic.

By the following lemma we show that the three-valued semantics for ATL* is a conservative extension of its two-valued semantics, as the two coincide whenever we consider standard iCGS with defined atoms.

Lemma 4.1.1 ([14]). *Let M be a standard iCGS, that is, $\mathcal{P}^{may} = \mathcal{P}^{must}$, $\delta^{may} = \delta^{must}$ are functions, and the truth value of every atom is defined (i.e., it is equal to either $\top$ or $\bot$). Then, for every formula ϕ in ATL*,*

$$((M, s) \models_3 \phi) = \top \quad \Leftrightarrow \quad (M, s) \models \phi \tag{4.1}$$

$$((M, s) \models_3 \phi) = \bot \quad \Leftrightarrow \quad (M, s) \not\models \phi \tag{4.2}$$

Thus, it immediately follows that model checking ATL* formulas under the three-valued semantics, with imperfect information and perfect recall is also undecidable.

However, for perfect information we can show the following.

Theorem 4.1.1 ([14]). *The model checking problem for generalized CGS (with perfect information) is* $2\mathrm{EXPTIME}$-COMPLETE *for ATL* and* PTIME-COMPLETE *for ATL.*

4.1.2 Abstraction

We now proceed to define perfect information, three-valued abstractions for iCGS. Subsequently, we demonstrate that defined truth values for ATL* formulas transfer from these abstractions to the original iCGS with imperfect information. Given that the model checking problem on the abstractions is decidable (as per Theorem 4.1.1), this preservation result can serve as the foundation for establishing a sound, albeit partial, verification procedure under imperfect information and perfect recall.

To begin with, given a coalition $A \subseteq Ag$ of agents, we define the *common knowledge relation* $\sim_A^C$ as the reflexive and transitive closure $(\bigcup_{i \in A} \sim_i)^*$ of the union of indistinguishability relations $\sim_i$ for $i \in A$ [6]. That is, $s \sim_A^C s'$ iff s' is reachable from s by a sequence $s_1, \ldots, s_n$ of states such that (i) $s_1 = s$, (ii) $s_n = s'$, and (iii) for every $j < n$, $s_j \sim_i s_{j+1}$ for some $i \in A$. Clearly, $\sim_A^C$ is an equivalence relation. Now, let $[s]_A = \{s' \in St \mid s' \sim_A s\}$ be the equivalence class of s according to $\sim_A$. The relation $\sim_A^C$ is extended to histories in a synchronous, pointwise way, *i.e.*, given $h, h' \in St^+$, $h \sim_A^C h'$ iff (i) $|h| = |h'|$ and (ii) for all $j \leq |h|$, $h_j \sim_A^C h'_j$. So, we introduce the notation $[h]_A = \{h' \in St^+ \mid h' \sim_A^C h\}$.

Definition 4.1.4. *Given an iCGS M and a coalition $A \subseteq Ag$, the* abstract (generalized) CGS *$M_A = \langle St_A, [s_I]_A, \{Act_i\}_{i \in Ag}, \mathcal{P}_A^{may}, \mathcal{P}_A^{must}, \delta_A^{may}, \delta_A^{must}, \mathcal{L}_A \rangle$ is defined as follows:*

1. *$St_A = \{[s]_A \mid s \in St\}$ is the set of equivalence classes for all states $s \in St$, with initial state $[s_I]_A$;*
2. *for every $t, t' \in St_A$ and joint action $\vec{\alpha}$, $t' \in \delta_A^{may}(t, \vec{\alpha})$ iff for some $s \in t$ and $s' \in t'$, $\delta(s, \vec{\alpha}) = s'$;*
3. *for every $t, t' \in St_A$ and joint action $\vec{\alpha}$, $t' \in \delta_A^{must}(t, \vec{\alpha})$ iff for all $s \in t$ there is $s' \in t'$ such that $\delta(s, \vec{\alpha}) = s'$;*
4. *for $x \in \{may, must\}$, $t \in St_A$, and $i \in Ag$, $\mathcal{P}_A^x(i, t) = \{\vec{\alpha}_i \in Act_i \mid \delta_A^x(t, (\vec{\alpha}_i, \vec{\alpha}_{\bar{i}}))$ is defined for some tuple of actions $\vec{\alpha}_{\bar{i}}\}$;*

5. *for* $\tau \in \{\top, \bot\}$*,* $a \in AP$*, and* $t \in St_A$*,* $\mathcal{L}_A(t, a) = \tau$ *iff* $\mathcal{L}(s, a) = \tau$ *for all* $s \in t$*; otherwise,* $\mathcal{L}_A(t, a) = \mathrm{uu}$*.*

We now show that the abstraction of an iCGS is indeed a generalized CGS as defined in Definition 4.1.1. In particular, the indistinguishability relation for every $i \in Ag$ is assumed to be the identity relation.

Lemma 4.1.2 ([14])**.** *For every coalition* $A \subseteq Ag$*, any abstraction* M_A *of an iCGS* M *is a generalized CGS.*

By next result, if an A-formula has a defined truth value in an abstract CGS M_A, built on an iCGS M, then the A-formula has the same truth value in M.

Theorem 4.1.2 ([14])**.** *Given an iCGS* M*, state* s*, and coalition* $A \subseteq Ag$*, for every* A*-formula* ϕ *in ATL*, we have that*

$$((M_A, [s]_A) \models_3 \phi) = \top \;\;\Rightarrow\;\; (M, s) \models \phi \tag{4.3}$$

$$((M_A, [s]_A) \models_3 \phi) = \bot \;\;\Rightarrow\;\; (M, s) \not\models \phi \tag{4.4}$$

4.1.3 Refinement

As stated in Theorem 4.1.2, if a formula is undefined on M_A, then no definitive conclusion can be drawn regarding the model checking problem for M. In this section, we provide a brief overview of the refinement procedure and its impact on the verification process, inspired by the concept of a "failure" state s_f as outlined in [14]. This procedure takes s_f as input and returns a refined CGS M^r_A. Intuitively, our procedure looks at incoming transitions into s_f. For concrete states s and s' in s_f, if the A-component of actions ending respectively in s and s' are different, any uniform strategy for A will visit either s or s'. As a result, the abstract state s_f can be split "safely" into an s- and an s'-component. We iterate this process until we analyze all possible relationships between concrete states in s_f with the goal of finding a partition that divides the failure state s_f in two new abstract states v and w while respecting the uniformity of the coalition A. If successful, the state space M^r_A is the state space M_A minus the failure state s_f plus the two new abstract states v and w created, that is $St^r_A = (St_A \setminus \{s_f\}) \cup \{v, w\}$. Given this new state space, we can construct the refined CGS M^r_A following the same construction rules as shown in Definition 4.1.4 for M_A. Note that if it is not possible to split the failure state s_f in a way that respects the uniformity of agents in the coalition A, our procedure cannot produce a refined CGS M^r_A different from M_A, and consequently terminates the verification process with an undefined value.

Given the high-level idea of our refinement procedure, we now show that must strategies respect uniformity *on the set of their outcomes* in refined CGS. First of all, note that given a path ρ in a refined CGS M^r_A, we can construct at least one path ρ' in its concrete CGS M such that for all $i \geq 1$, $\rho'_i \in \rho_i$. Since we could potentially construct multiple paths from ρ, we denote by $\rho' \in \rho$ a concrete path ρ' from ρ. Now, we have all the ingredients to present the following result.

Lemma 4.1.3 ([14])**.** *Given a refined CGS* M^r_A*, for every joint strategy* σ^{must}_A*, for all* $\rho, \hat{\rho} \in out(t, \sigma^{must}_A)$*, all* $\rho' \in \rho$*,* $\hat{\rho}' \in \hat{\rho}$*, and all* $i \in A$*,* $j \in \mathbb{N}$*, if* $\rho'_{\leq j} \sim_i \hat{\rho}'_{\leq j}$ *then* $\sigma^{must}_i(\rho_{\leq j}) = \sigma^{must}_i(\hat{\rho}_{\leq j})$*.*

By Lemma 4.1.3 we can prove the main preservation result of this section. In particular, the lemma is used in the inductive step for strategy operators.

Theorem 4.1.3 ([14]). *Given an iCGS M, state s, coalition A, its abstract CGS M_A with refinement M_A^r, and state $s_A^r \ni s$, for every A-formula ϕ in ATL*,*

$$((M_A^r, s_A^r) \models_3 \phi) = \top \quad \Rightarrow \quad (M, s) \models \phi \tag{4.5}$$

$$((M_A^r, s_A^r) \models_3 \phi) = \bot \quad \Rightarrow \quad (M, s) \not\models \phi \tag{4.6}$$

Then, we can conclude with the complexity of our procedure.

Theorem 4.1.4 ([14]). *Our procedure terminates in* $2\mathrm{EXPTIME}$ *if* φ *is in ATL*, in* PTIME *if* φ *is in ATL.*

It is important to note that our procedure may not always terminate with a defined truth value. The model checking problem for ATL (and by extension, for ATL*) in the context of imperfect information and perfect recall is undecidable in general. Thus, our procedure is designed to be a sound, albeit partial, verification algorithm.

4.2 Approximate the memory of the strategies

In this contribution, we introduce a novel three-valued semantics for ATL* under bounded recall, which encompasses perfect recall as well as a limit case. We investigate the corresponding model checking problem and explore the formal properties of three-valued ATL* in comparison to the traditional, two-valued semantics. Our key finding is that bounded recall serves as a provably sound approximation of perfect recall in terms of verification. Building upon these theoretical findings, we establish the groundwork for a verification procedure for model checking MAS under imperfect information and perfect recall. This procedure involves iteratively checking bounded recall versions of the same MAS in the three-valued semantics, with increasing levels of memory. While the algorithm may not provide complete results in all cases, we demonstrate that assuming a bound on recall enables termination within $\mathrm{EXPTIME}$.

First of all, given $St^{<n}$ representing the set of histories of length less than or equal to n, we provide the formal definition of bounded recall strategy.

Definition 4.2.1. *For $n \in \mathbb{N}^+ \cup \{\omega\}$, a* uniform strategy with n-bounded recall *for agent $i \in Ag$ is a function $\sigma_i^n : St^{<1+n} \to Act_i$ such that for all histories $h, h' \in St^{<1+n}$, (i) $\sigma_i^n(h) \in \mathcal{P}(i, last(h))$; and (ii) $h \sim_i h'$ implies $\sigma_i^n(h) = \sigma_i^n(h')$.*

The semantics of ATL* with n-bounded recall strategies is the same as in Definition 1.2.4 but replacing "$\models$" with "$\models^n$" and "strategy" with "n-bounded recall strategy".

Now, we show the model checking problem for bounded recall within the two-valued semantics, defined as follows.

Definition 4.2.2. *The* bounded model checking problem *concerns determining whether, given an iCGS M, ATL* formula ϕ, bound $n \in \mathbb{N}^+ \cup \{\omega\}$, truth value $\tau \in \{\top, \bot\}$, it is the case that $(M \models^n \phi) = \tau$.*

Based on the definition provided above, in [10, 11], we establish that model checking ATL* with perfect recall (i.e., for $n = \omega$) and imperfect information is undecidable. Conversely, model checking ATL* with bounded recall and imperfect information is shown to be decidable. This decidability result forms the foundation for a partial model checking procedure for perfect recall, which involves incrementally increasing the bound n on the memory of agents. However, as demonstrated below, increasing memory only preserves relatively limited fragments of ATL* and may, therefore, be of limited interest.

Lemma 4.2.1 ([11]). *Let $m, n \in \mathbb{N}^+ \cup \{\omega\}$ be such that $m \leq n$; let ψ be an existential and ϕ an universal formula in ATL*. Then,*

$$(M, s) \models^m \psi \quad \Rightarrow \quad (M, s) \models^n \psi \tag{4.7}$$
$$(M, s) \not\models^m \phi \quad \Rightarrow \quad (M, s) \not\models^n \phi \tag{4.8}$$

By Lemma 4.2.1 adding memory preserves the truth of existential formulas as well as falsehood of universal formulas. However, it is not difficult to find counterexamples to the extensions of (4.7) and (4.8) even in ATL.

Lemma 4.2.2 ([11]). *Let $m, n \in \mathbb{N}^+ \cup \{\omega\}$ be such that $m < n$. There exists formulas φ and $\varphi' = \neg\varphi$ in ATL such that*

$$(M, s) \not\models^m \varphi \quad \text{and} \quad (M, s) \models^n \varphi \tag{4.9}$$
$$(M, s) \models^m \varphi' \quad \text{and} \quad (M, s) \not\models^n \varphi' \tag{4.10}$$

By Lemmas 4.2.1 and 4.2.2 any naive attempt to approximate perfect recall by increasing bounded recall is severely restricted in two ways. Firstly, Lemma 4.2.1 holds only for the existential and universal fragments of ATL*. Secondly, only the truth of existential formulas is preserved by adding memory, whereas negative results can only be lifted for the universal fragment. For this reason, we introduce a three-valued semantics to overcome these difficulties.

Definition 4.2.3. *Let $n \in \mathbb{N}^+ \cup \{\omega\}$. The three-valued satisfaction relation $\models_3^n$ for an iCGS M, state s, path ρ, ATL* formula ϕ, and $\tau \in \{\top, \bot\}$ is defined as follows, where $\neg\top = \bot$ and $\neg\bot = \top$:*

$((M, s) \models_3^n a) = \tau$	*iff*	$\mathcal{L}(s, a) = \tau$
$((M, s) \models_3^n \neg\varphi) = \tau$	*iff*	$((M, s) \models_3^n \varphi) = \neg\tau$
$((M, s) \models_3^n \varphi \wedge \varphi') = \top$	*iff*	$((M, s) \models_3^n \varphi) = \top$ *and* $((M, s) \models_3^n \varphi') = \top$
$((M, s) \models_3^n \varphi \wedge \varphi') = \bot$	*iff*	$((M, s) \models_3^n \varphi) = \bot$ *or* $((M, s) \models_3^n \varphi') = \bot$
$((M, s) \models_3^n \langle\langle A\rangle\rangle\psi) = \top$	*iff*	*for some* σ_A^n, *for all* $\rho \in out(s, \sigma_A^n)$, $((M, \rho) \models_3^n \psi) = \top$
$((M, s) \models_3^n \langle\langle A\rangle\rangle\psi) = \bot$	*iff*	*for some* $\sigma_{\bar{A}}^n$, *for all* $\rho \in out(s, \sigma_{\bar{A}}^n)$, $((M, \rho) \models_3^n \psi) = \bot$
$((M, \rho) \models_3^n \varphi) = \tau$	*iff*	$((M, \rho_1) \models_3^n \varphi) = \tau$
$((M, \rho) \models_3^n \neg\psi) = \tau$	*iff*	$((M, \rho) \models_3^n \psi) = \neg\tau$
$((M, \rho) \models_3^n \psi \wedge \psi') = \top$	*iff*	$((M, \rho) \models_3^n \psi) = \top$ *and* $((M, \rho) \models_3^n \psi') = \top$
$((M, \rho) \models_3^n \psi \wedge \psi') = \bot$	*iff*	$((M, \rho) \models_3^n \psi) = \bot$ *or* $((M, \rho) \models_3^n \psi') = \bot$
$((M, \rho) \models_3^n X\psi) = \tau$	*iff*	$((M, \rho_{\geq 2}) \models_3^n \psi) = \tau$
$((M, \rho) \models_3^n \psi U \psi') = \top$	*iff*	*for some* $k \geq 1$, $((M, \rho_{\geq k}) \models_3^n \psi') = \top$, *and* *for all* j, $1 \leq j < k$ *implies* $((M, \rho_{\geq j}) \models_3^n \psi) = \top$
$((M, \rho) \models_3^n \psi U \psi') = \bot$	*iff*	*for all* $k \geq 1$, *either* $((M, \rho_{\geq k}) \models_3^n \psi') = \bot$ *or for some* j, $1 \leq j < k$ *and* $((M, \rho_{\geq j}) \models_3^n \psi) = \bot$

In all other cases the value of ϕ is undefined (uu).

As for the two-valued case, in [10, 11] we show that model checking ATL* for tree-valued semantics with perfect recall (i.e., for $n = \omega$) and imperfect information is undecidable while model checking ATL* with bounded recall and imperfect information is decidable. Our aim in the rest of this section is to lay the theoretical foundations of a (partial) model checking procedure that is able to deal with the whole of ATL*. To this end, the next result, which is akin to Lemma 4.2.1, details the preservation of ATL* formulas when adding memory. However, differently from Lemma 4.2.1, this result holds for all ATL* formulas.

Lemma 4.2.3. *Let $m, n \in \mathbb{N}^+ \cup \{\omega\}$ be such that $m \leq n$; let ϕ be a formula in ATL*. Then,*

$$((M, s) \models_3^m \phi) = \top \;\Rightarrow\; ((M, s) \models_3^n \phi) = \top \tag{4.11}$$

$$((M, s) \models_3^m \phi) = \bot \;\Rightarrow\; ((M, s) \models_3^n \phi) = \bot \tag{4.12}$$

By Lemma 4.2.3 adding memory preserves defined truth values for all formulas in ATL*. This is in marked contrast with Lemma 4.2.1. Indeed, even though in some cases the value of an ATL* formula may be undefined in the three-valued semantics, whenever it is defined, it does not change if memory is added.

Now, we have all the elements to conclude our main result on the relationship between bounded recall and the two-and three-valued semantics.

Lemma 4.2.4. *Let $m, n \in \mathbb{N}^+ \cup \{\omega\}$ be such that $m \leq n$; let ϕ be a formula in ATL*. Then,*

$$((M, s) \models_3^m \phi) = \top \;\Rightarrow\; (M, s) \models^n \phi \tag{4.13}$$

$$((M, s) \models_3^m \phi) = \bot \;\Rightarrow\; (M, s) \models^n \phi \tag{4.14}$$

We can conclude this section with a partial decision procedure for model checking ATL* under the assumptions of imperfect information and n-bounded recall. It is partial, as it is not guaranteed to terminate for the case of perfect recall, that is, for $n = \omega$. This procedure is described in algorithm `Iterative_MC`(M, ϕ, n).

Algorithm 1 `Iterative_MC`(M, ϕ, n)

1: $j = 1$, $k = \mathrm{uu}$;
2: **while** $j \leq n$ and $k = \mathrm{uu}$ **do**
3:     **if** `MC3`$(M, \phi, j, \top)$ **then** $k = \top$
4:     **else if** `MC3`$(M, \phi, j, \bot)$ **then** $k = \bot$
5:     **end if**
6:     $j = j + 1$;
7: **end while**
8: **if** $k \neq \mathrm{uu}$ **then return** (j, k);
9: **else return** -1;
10: **end if**

It takes as input an iCGS M, an ATL* formula ϕ, and a bound $n \in \mathbb{N}^+ \cup \{\omega\}$. It comprises a `while`-loop (lines 2-7), which checks whether the bound has not yet been reached ($j < n$) and ϕ has not yet been decided ($k = \mathrm{uu}$). Within the loop, the formula ϕ is model-checked in M according to the three-valued semantics using the subroutine `MC3`$()$, and the result is stored in variable k. Upon exiting the loop, variable k is examined (line 8). If $k \neq \mathrm{uu}$, it indicates that the loop was exited due to a defined answer for the three-valued model checking problem with j-bounded recall (and possibly the bound n was reached). By Lemma 4.2.4, we can then transfer the returned value to the corresponding model checking problem for the two-valued

Algorithm 2 `MC3`$(M, \langle\langle A\rangle\rangle\phi, n, \tau)$

if $\tau = \top$ **then return** $M \models^n \langle\langle A\rangle\rangle\phi$
else if $\tau = \bot$ **then return** $M \models^n \langle\langle \bar{A}\rangle\rangle\neg\phi$
else
  if $\tau = \mathrm{uu}$ and $M \models^n \langle\langle A\rangle\rangle\phi \vee M \models^n \langle\langle \bar{A}\rangle\rangle\neg\phi$ **then return** $\bot$
  else return $\top$
  end if
end if

semantics. Conversely, if $k = \mathrm{uu}$, it signifies that the bound has been reached in the loop, and the default value -1 is returned to indicate termination without a defined truth value. We will now demonstrate the termination of the algorithm for $n \in \mathbb{N}^+$, as well as its soundness.

Theorem 4.2.1 ([11]). *For $n \in \mathbb{N}^+$, `Iterative_MC()` terminates in EXPTIME. Moreover, `Iterative_MC()` is sound: if the value returned is different from -1, then $M \models^n \phi$ iff $k = \top$ and $M \not\models^n \phi$ iff $k = \bot$.*

An important application of `Iterative_MC()` is for $n = \omega$, namely model checking perfect recall. In such a case termination is no longer guaranteed, but soundness is.

Theorem 4.2.2 ([11]). *For $n = \omega$, `Iterative_MC()` does not necessarily terminate. However, `Iterative_MC()` is sound: if the value returned is different from -1, then $M \models^n \phi$ iff $k = \top$ and $M \not\models^n \phi$ iff $k = \bot$.*

4.3 Approximate the model and logic

In this section, we introduce another technique to approximate the verification of ATL* under imperfect information and perfect recall, a problem known to be undecidable. Our approach involves generating sub-models of the original model M, wherein each sub-model M' satisfies a sub-formula φ' of φ, and the verification of φ' in M' is decidable. We then leverage CTL* model checking to obtain a verification result of φ on M. We demonstrate that our procedure is sound and shares the same complexity class as ATL* model checking under perfect information and perfect recall.

The idea behind sub-models lies in their ability to capture both under-approximations and over-approximations of the original model M. Specifically, each negative sub-model M^n serves as an under-approximation, while positive sub-model M^p serves as an over-approximation of M. Consequently, negative sub-models can be employed to establish the satisfaction of properties, whereas positive sub-models can help identify property violations. Therefore, if we manage to find a strategy to satisfy a property in the negative sub-model, it implies that the same strategy can be utilized in the original model. Conversely, if we fail to find a strategy in the positive sub-model, it indicates that no strategy exists in the original model.

Definition 4.3.1. *Given an iCGS M and $x \in \{n, p\}$, we denote with $M^x = \langle St^x, s_I^x, \{Act_i\}_{i \in Ag}, \{\sim_i^n\}_{i \in Ag}, \mathcal{P}^n, \delta^n, \mathcal{L}^n\rangle$ a sub-model of M such that:*

- *The set of states is defined as $St^x = St^\star \cup \{s_x\}$, where $St^\star \subseteq St$, and $s_I^x \in St^\star$ is the initial state.*
- *$\sim_i^x$ is defined as the corresponding $\sim_i$ restricted to $St^\star$.*

- *The protocol function is defined as* $\mathcal{P}^x : Ag \times St^x \to (2^{Act} \setminus \{\emptyset\})$, *where* $\mathcal{P}^x(i, s) = \mathcal{P}(i, s)$, *for every* $s \in St^\star$ *and* $\mathcal{P}^x(i, s_x) = Act_i$, *for all* $i \in Ag$.
- *The transition function is defined as* $\delta^x : St^x \times ACT \to St^x$, *where given a transition* $\delta(s, \vec{\alpha}) = s'$, *if* $s, s' \in St^\star$ *then* $\delta^x(s, \vec{\alpha}) = \delta(s, \vec{\alpha}) = s'$ *else if* $s' \in St \setminus St^\star$ *and* $s \in St^x$ *then* $\delta^x(s, \vec{\alpha}) = s_x$.
- *For all* $s \in St^\star$, $\mathcal{L}^x(s) = \mathcal{L}(s)$, *and if* $x = n$ *then* $\mathcal{L}^n(s_n) = \emptyset$, *otherwise* $\mathcal{L}^p(s_p) = AP$.

Informally, s_x *is a sink state that replaces the states removed from the original model (i.e.,* $St \setminus St^\star$*).*

Given the definition above, we now show a preservation result from our sub-models to the original model that we will use in our partial procedure.

Lemma 4.3.1 ([16])**.** *Given a model* M*, a negative (resp., positive) sub-model with perfect information* M^n *(resp.,* M^p*) of* M*, and a formula* φ *of the form* $\varphi = \langle\langle A \rangle\rangle \psi$ *for some* $A \subseteq Ag$*. For any* $s \in St^\star$*, we have that:*

$$M^n, s \models \varphi \Rightarrow M, s \models \varphi$$
$$M^p, s \not\models \varphi \Rightarrow M, s \not\models \varphi$$

Before providing the idea of our procedure, we also recall a result derived from [5] that we use to approximate the logic.

Lemma 4.3.2 ([16])**.** *Given a model* M*, a formula* φ *in ATL* written in Negation Normal Form (NNF)* [1]*, and the CTL* universal* [2] *(resp., existential* [3]*) version* φ_A *(resp.,* φ_E*) of* φ*. For any* $s \in St$*, we have that:*

$$M, s \models \varphi_A \Rightarrow M, s \models \varphi$$
$$M, s \not\models \varphi_E \Rightarrow M, s \not\models \varphi$$

Given the above results, we are able to show our procedure.

Algorithm 3 `ModelCheckingProcedure`(M, φ)

```
Preprocessing(M, φ)
candidates = FindSubModels(M, φ)
if |candidates| = 1 then return M ⊨ φ
end if
while candidates is not empty do
    extract ⟨M^n, M^p⟩ from candidates
    result = CheckSubFormulas(⟨M^n, M^p⟩, φ)
    k = Verification(M, φ, result)
    if k ≠ ? then return k
    end if
end while
return ?
```

The `ModelCheckingProcedure()` takes as input a model M and a formula φ. It begins by calling the `Preprocessing()` function to generate the NNF of φ and replace all negated

[1] It is an equivalent version of ATL* in which the negation is used only in front of atoms.
[2] It is the version in which we replace every ATL* strategic operator with a CTL* universal operator.
[3] It is the version in which we replace every ATL* strategic operator with a CTL* existential operator.

atoms with new positive atoms inside both M and φ. Then, it invokes the `FindSubModels()` function to generate all positive and negative sub-models representing all possible sub-models with perfect information. If the number of candidates is equal to one (line 3), indicating that the input model M has perfect information, the procedure directly calls the ATL* model checking procedure for perfect information and perfect recall.

Algorithm 4 `Verification` (M, φ, $result$)

1: $k = ?$
2: **for** $s \in St$ **do**
3: take set $atoms$ from $result(s)$
4: `UpdateModel`(M, s, $atoms$)
5: **end for**
6: $\varphi_n = \varphi$, $\varphi_p = \varphi$
7: **while** $result$ is not empty **do**
8: extract $\langle s, \psi, vatom_\psi \rangle$ from $result$
9: **if** $v = n$ **then** $\varphi_n =$ `UpdateFormula`(ψ_n, ψ, $natom_\psi$)
10: **else** $\varphi_p =$ `UpdateFormula`(ψ_p, ψ, $patom_\psi$)
11: **end if**
12: **end while**
13: $\varphi_A =$ `FromATLtoCTL`(φ_n, n)
14: $\varphi_E =$ `FromATLtoCTL`(φ_p, p)
15: **if** $M \models \varphi_A$ **then** $k = \top$
16: **end if**
17: **if** $M \not\models \varphi_E$ **then** $k = \bot$
18: **end if**
19: **return** k

Subsequently, a while loop (lines 5-11) iterates through each candidate, checking the sub-formulas of φ true on the sub-models of M via `CheckSubFormulas()`, and assessing the truth value of the entire formula via `Verification()`. The latter procedure utilizes the idea of the bottom-up approach, combining the results obtained on different sub-formulas and propagating them onto the original model, which is made valid by Lemma 4.3.1. Finally, it approximates verification with CTL* using the result from Lemma 4.3.2. If the output of the latter procedure is different from ?, indicating a defined result, it is returned directly (line 9).

We conclude with two results on the complexity and soundness of the proposed approach.

Theorem 4.3.1 ([16]). *Algorithm 3 terminates in double exponential time w.r.t. the size of φ and exponential time w.r.t. the size of M.*

Theorem 4.3.2 ([16]). *Algorithm 3 is sound: if the value returned is not ?, then $M \models \varphi$ iff $k = \top$ and $M \not\models \phi$ iff $k = \bot$.*

Chapter 5

Conclusion and Future directions

In this document, we have presented the main works in the field of multi-agent system verification in which we have collaborated over the past decade. In particular, we have divided these works into three major categories: modeling, specification, and verification of multi-agent systems. For each of these macro areas, we have dedicated a chapter. In Chapter 2, we have discussed a "natural" method for modeling strategies. We have presented these strategies in the context of perfect and imperfect information and presented model checking results. In Chapter 3, we have illustrated logics that allow specifying properties in which there are agents capable of modifying the MAS model. This last aspect is extremely important in the context of cybersecurity, as illustrated in our exemplifying use case. Furthermore, we have shown that these logics are more expressive than CTL and ATL while the model checking complexity remains polynomial. Finally, in Chapter 4, we have shown that model checking for some sub-classes of MAS in the context of imperfect information and memoryfull strategies, a problem that is undecidable in general, is decidable. In particular, we have defined approximation methods on information, memory of strategies, and model topology and specification, and provided preservation results.

The choice of the content of the three technical chapters was made considering not only the quality of the publications produced but also to present the topics in which we have been most involved throughout our research journey. In particular, the first work on natural strategies (the subject of Chapter 2) was developed and published during a visit to the IPI PAN research center in Poland during our PhD at the University of Naples, and it is still a very fertile topic that has led to further publications in recent years and will lead to more in the near future. The entire scope related to finding decidability in generally undecidable problems (the subject of Chapter 4) is part of a path that we started during our postdoc at the University of Evry and continued when we acquired the position of associate professor at Telecom Paris. Finally, we presented the results obtained with our first postdoc, within a collaboration at Telecom Paris, concerning logics for cybersecurity (the subject of Chapter 3), to showcase a research line that has emerged in recent years and could have a significant impact in the next decade, and especially to demonstrate the important results obtained in the role of supervisors. In addition to these three research lines that will be part of our future work in the short and medium term, we also briefly present other research objectives in which we will be involved:

- One of the main objectives in the near future will be to participate in national and international projects as a principal investigator or partner. From this perspective, two projects have been submitted this year with the role of principal investigator: ANR JCJC and ERC Starting Grant. We are waiting the final verdict for both of them. Additionally, we have submitted an ANR PRC project, and we are currently working on a Horizon project as a partner.

- We are developing a new compositional framework for model checking multi-agent systems called VITAMIN [105] that can be used by any user, expert and non-expert of formal verification. To accelerate the development of the tool, we have submitted some projects on this subject and we have filed two patents. We also have a prototype of tool that has been implemented during the last two years.

- We have started, with a PhD student (in collaboration with the company EDF), an in-depth study of cybersecurity risks. In particular, we are studying new formal methods techniques to avoid cyberattacks. To tackle these issues we are working on a logic to identify agents capabilities. The first work on this field has been accepted some months ago [106].

- Related to the latter direction, we will open a new PhD position to synthesize strategies in the context of cyberdefense by following the ideas in [50, 51].

- In the light of [59], we are working to runtime solutions in distributed and competitive monitors with imperfect information. We think this topic could be of interest not only for the research community but also in the industrial context.

- Finally, another line of research we are involved in, which we believe could be as relevant for the scientific and industrial community as the cybersecurity topic, focuses on the study of model checking techniques to verify the correctness of smart contracts within blockchain contexts.

Bibliography

[1] E.M. Clarke, O. Grumberg, D. Peled, and D.A. Peled. *Model Checking*. The Cyber-Physical Systems Series. MIT Press, 1999.

[2] E.M. Clarke, O. Grumberg, D. Kroening, D. Peled, and H. Veith. *Model Checking, second edition*. Cyber Physical Systems Series. MIT Press, 2018.

[3] Amir Pnueli. The temporal logic of programs. In *18th Annual Symposium on Foundations of Computer Science, Providence, Rhode Island, USA, 31 October - 1 November 1977*, pages 46–57. IEEE Computer Society, 1977.

[4] E. Allen Emerson and Edmund M. Clarke. Using branching time temporal logic to synthesize synchronization skeletons. *Sci. Comput. Program.*, 2(3):241–266, 1982.

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

[6] Ronald Fagin, Joseph Y. Halpern, Yoram Moses, and Moshe Y. Vardi. *Reasoning About Knowledge*. MIT Press, Cambridge, MA, USA, 2003.

[7] Fabio Mogavero, Aniello Murano, Giuseppe Perelli, and Moshe Y. Vardi. Reasoning about strategies: On the model-checking problem. *ACM Trans. Comput. Log.*, 15(4):34:1–34:47, 2014.

[8] Pierre-Yves Schobbens. Alternating-time logic with imperfect recall. In Wiebe van der Hoek, Alessio Lomuscio, Erik P. de Vink, and Michael J. Wooldridge, editors, *1st International Workshop on Logic and Communication in Multi-Agent Systems, LCMAS 2003, Eindhoven, The Netherlands, June 29, 2003*, volume 85 of *Electronic Notes in Theoretical Computer Science*, pages 82–93. Elsevier, 2003.

[9] Catalin Dima and Ferucio Laurentiu Tiplea. Model-checking ATL under imperfect information and perfect recall semantics is undecidable. *CoRR*, abs/1102.4225, 2011.

[10] Francesco Belardinelli, Alessio Lomuscio, and Vadim Malvone. Approximating perfect recall when model checking strategic abilities. In Michael Thielscher, Francesca Toni, and Frank Wolter, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the Sixteenth International Conference, KR 2018, Tempe, Arizona, 30 October - 2 November 2018*, pages 435–444. AAAI Press, 2018.

[11] Francesco Belardinelli, Alessio Lomuscio, Vadim Malvone, and Emily Yu. Approximating perfect recall when model checking strategic abilities: Theory and applications. *J. Artif. Intell. Res.*, 73:897–932, 2022.

[12] Francesco Belardinelli, Alessio Lomuscio, and Vadim Malvone. An abstraction-based method for verifying strategic properties in multi-agent systems with imperfect information. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6030–6037. AAAI Press, 2019.

[13] Francesco Belardinelli and Vadim Malvone. Verifying strategic abilities in multi-agent systems via first-order entailment. In Giuseppe De Giacomo, Alejandro Catalá, Bistra Dilkina, Michela Milano, Senén Barro, Alberto Bugarín, and Jérôme Lang, editors, *ECAI 2020 - 24th European Conference on Artificial Intelligence, 29 August-8 September 2020, Santiago de Compostela, Spain, August 29 - September 8, 2020 - Including 10th Conference on Prestigious Applications of Artificial Intelligence (PAIS 2020)*, volume 325 of *Frontiers in Artificial Intelligence and Applications*, pages 27–34. IOS Press, 2020.

[14] Francesco Belardinelli, Angelo Ferrando, and Vadim Malvone. An abstraction-refinement framework for verifying strategic properties in multi-agent systems with imperfect information. *Artif. Intell.*, 316:103847, 2023.

[15] Angelo Ferrando and Vadim Malvone. Towards the combination of model checking and runtime verification on multi-agent systems. In Frank Dignum, Philippe Mathieu, Juan Manuel Corchado, and Fernando de la Prieta, editors, *Advances in Practical Applications of Agents, Multi-Agent Systems, and Complex Systems Simulation. The PAAMS Collection - 20th International Conference, PAAMS 2022, L'Aquila, Italy, July 13-15, 2022, Proceedings*, volume 13616 of *Lecture Notes in Computer Science*, pages 140–152. Springer, 2022.

[16] Angelo Ferrando and Vadim Malvone. Towards the verification of strategic properties in multi-agent systems with imperfect information. In Noa Agmon, Bo An, Alessandro Ricci, and William Yeoh, editors, *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2023, London, United Kingdom, 29 May 2023 - 2 June 2023*, pages 793–801. ACM, 2023.

[17] Bastien Maubert, Munyque Mittelmann, Aniello Murano, and Laurent Perrussel. Strategic reasoning in automated mechanism design. In Meghyn Bienvenu, Gerhard Lakemeyer, and Esra Erdem, editors, *Proceedings of the 18th International Conference on Principles of Knowledge Representation and Reasoning, KR 2021, Online event, November 3-12, 2021*, pages 487–496, 2021.

[18] Francesco Belardinelli, Wojciech Jamroga, Damian Kurpiewski, Vadim Malvone, and Aniello Murano. Strategy logic with simple goals: Tractable reasoning about strategies. In Sarit Kraus, editor, *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 88–94. ijcai.org, 2019.

[19] Arnaud Da Costa Lopes, François Laroussinie, and Nicolas Markey. ATL with strategy contexts: Expressiveness and model checking. In Kamal Lodaya and Meena Mahajan,

editors, *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2010, December 15-18, 2010, Chennai, India*, volume 8 of *LIPIcs*, pages 120–132. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2010.

[20] Alessio Lomuscio and Jakub Michaliszyn. Verifying multi-agent systems by model checking three-valued abstractions. In Gerhard Weiss, Pinar Yolum, Rafael H. Bordini, and Edith Elkind, editors, *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2015, Istanbul, Turkey, May 4-8, 2015*, pages 189–198. ACM, 2015.

[21] Wojciech Jamroga, Vadim Malvone, and Aniello Murano. Reasoning about natural strategic ability. In Kate Larson, Michael Winikoff, Sanmay Das, and Edmund H. Durfee, editors, *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2017, São Paulo, Brazil, May 8-12, 2017*, pages 714–722. ACM, 2017.

[22] Wojciech Jamroga, Vadim Malvone, and Aniello Murano. Natural strategic ability. *Artif. Intell.*, 277, 2019.

[23] Wojciech Jamroga, Vadim Malvone, and Aniello Murano. Natural strategic ability under imperfect information. In Edith Elkind, Manuela Veloso, Noa Agmon, and Matthew E. Taylor, editors, *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '19, Montreal, QC, Canada, May 13-17, 2019*, pages 962–970. International Foundation for Autonomous Agents and Multiagent Systems, 2019.

[24] Francesco Belardinelli, Wojtek Jamroga, Vadim Malvone, Munyque Mittelmann, Aniello Murano, and Laurent Perrussel. Reasoning about human-friendly strategies in repeated keyword auctions. In Piotr Faliszewski, Viviana Mascardi, Catherine Pelachaud, and Matthew E. Taylor, editors, *21st International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2022, Auckland, New Zealand, May 9-13, 2022*, pages 62–71. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS), 2022.

[25] Vadim Malvone. The impact of strategies and information in model checking for multi-agent systems. In Angelo Ferrando and Rafael Cardoso, editors, *Proceedings of the Third Workshop on Agents and Robots for reliable Engineered Autonomy, AREA@ECAI 2023, Krakow, Poland, 1st October 2023*, volume 391 of *EPTCS*, pages 63–70, 2023.

[26] Gabriel Ballot, Vadim Malvone, Jean Leneutre, and Etienne Borde. Reasoning about moving target defense in attack modeling formalisms. In Hamed Okhravi and Cliff Wang, editors, *Proceedings of the 9th ACM Workshop on Moving Target Defense, MTD 2022, Los Angeles, CA, USA, 7 November 2022*, pages 55–65. ACM, 2022.

[27] Davide Catta, Jean Leneutre, and Vadim Malvone. Towards a formal verification of attack graphs. In Riccardo De Benedictis, Nicola Gatti, Marco Maratea, Andrea Micheli, Aniello Murano, Enrico Scala, Luciano Serafini, Ivan Serina, Alessandro Umbrico, and Mauro Vallati, editors, *Proceedings of the 10th Italian workshop on Planning and Scheduling (IPS 2022), RCRA Incontri E Confronti (RiCeRcA 2022), and the workshop on Strategies, Prediction, Interaction, and Reasoning in Italy (SPIRIT 2022) co-located*

with 21st International Conference of the Italian Association for Artificial Intelligence (AIxIA 2022), November 28 - December 2, 2022, University of Udine, Udine, Italy, volume 3345 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2022.

[28] Davide Catta, Antonio Di Stasio, Jean Leneutre, Vadim Malvone, and Aniello Murano. A game theoretic approach to attack graphs. In Ana Paula Rocha, Luc Steels, and H. Jaap van den Herik, editors, *Proceedings of the 15th International Conference on Agents and Artificial Intelligence, ICAART 2023, Volume 1, Lisbon, Portugal, February 22-24, 2023*, pages 347–354. SCITEPRESS, 2023.

[29] Davide Catta, Jean Leneutre, Antonina Mijatovic, Johanna Ulin, and Vadim Malvone. A formal verification approach to handle attack graphs. In Ana Paula Rocha, Luc Steels, and H. Jaap van den Herik, editors, *Proceedings of the 16th International Conference on Agents and Artificial Intelligence, ICAART 2024, Volume 3, Rome, Italy, February 24-26, 2024*, pages 125–132. SCITEPRESS, 2024.

[30] Vadim Malvone, Aniello Murano, and Loredana Sorrentino. Games with additional winning strategies. In Davide Ancona, Marco Maratea, and Viviana Mascardi, editors, *Proceedings of the 30th Italian Conference on Computational Logic, Genova, Italy, July 1-3, 2015*, volume 1459 of *CEUR Workshop Proceedings*, pages 175–180. CEUR-WS.org, 2015.

[31] Vadim Malvone and Aniello Murano. Additional winning strategies in two-player games. In Vittorio Bilò and Antonio Caruso, editors, *Proceedings of the 17th Italian Conference on Theoretical Computer Science, Lecce, Italy, September 7-9, 2016*, volume 1720 of *CEUR Workshop Proceedings*, pages 251–256. CEUR-WS.org, 2016.

[32] Vadim Malvone, Aniello Murano, and Marco Tafuto. Nwin: A tool for counting winning strategies (demonstration). In Catholijn M. Jonker, Stacy Marsella, John Thangarajah, and Karl Tuyls, editors, *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems, Singapore, May 9-13, 2016*, pages 1501–1503. ACM, 2016.

[33] Vadim Malvone and Aniello Murano. Reasoning about additional winning strategies in two-player games. In Francesco Belardinelli and Estefania Argente, editors, *Multi-Agent Systems and Agreement Technologies - 15th European Conference, EUMAS 2017, and 5th International Conference, AT 2017, Évry, France, December 14-15, 2017, Revised Selected Papers*, volume 10767 of *Lecture Notes in Computer Science*, pages 163–171. Springer, 2017.

[34] Vadim Malvone, Aniello Murano, and Loredana Sorrentino. Additional winning strategies in reachability games. *Fundam. Informaticae*, 159(1-2):175–195, 2018.

[35] Vadim Malvone, Aniello Murano, and Loredana Sorrentino. Hiding actions in concurrent games. In Gal A. Kaminka, Maria Fox, Paolo Bouquet, Eyke Hüllermeier, Virginia Dignum, Frank Dignum, and Frank van Harmelen, editors, *ECAI 2016 - 22nd European Conference on Artificial Intelligence, 29 August-2 September 2016, The Hague, The Netherlands - Including Prestigious Applications of Artificial Intelligence (PAIS 2016)*, volume 285 of *Frontiers in Artificial Intelligence and Applications*, pages 1686–1687. IOS Press, 2016.

[36] Vadim Malvone, Aniello Murano, and Loredana Sorrentino. Hiding actions in multi-player games. In Kate Larson, Michael Winikoff, Sanmay Das, and Edmund H. Durfee, editors, *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2017, São Paulo, Brazil, May 8-12, 2017*, pages 1205–1213. ACM, 2017.

[37] Francesco Belardinelli, Ioana Boureanu, Catalin Dima, and Vadim Malvone. Verifying strategic abilities in multi-agent systems with private data-sharing. In Edith Elkind, Manuela Veloso, Noa Agmon, and Matthew E. Taylor, editors, *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '19, Montreal, QC, Canada, May 13-17, 2019*, pages 1820–1822. International Foundation for Autonomous Agents and Multiagent Systems, 2019.

[38] Francesco Belardinelli, Ioana Boureanu, Catalin Dima, and Vadim Malvone. Model checking strategic abilities in information-sharing systems. *CoRR*, abs/2204.08896, 2022.

[39] Vadim Malvone, Aniello Murano, and Loredana Sorrentino. Concurrent multi-player parity games. In Catholijn M. Jonker, Stacy Marsella, John Thangarajah, and Karl Tuyls, editors, *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems, Singapore, May 9-13, 2016*, pages 689–697. ACM, 2016.

[40] Wojciech Jamroga, Damian Kurpiewski, and Vadim Malvone. Natural strategic abilities in voting protocols. In Thomas Groß and Luca Viganò, editors, *Socio-Technical Aspects in Security and Trust - 10th International Workshop, STAST 2020, Virtual Event, September 14, 2020, Revised Selected Papers*, volume 12812 of *Lecture Notes in Computer Science*, pages 45–62. Springer, 2020.

[41] Wojciech Jamroga, Damian Kurpiewski, and Vadim Malvone. How to measure usable security: Natural strategies in voting protocols. *J. Comput. Secur.*, 30(3):381–409, 2022.

[42] Vadim Malvone, Fabio Mogavero, Aniello Murano, and Loredana Sorrentino. On the counting of strategies. In Fabio Grandi, Martin Lange, and Alessio Lomuscio, editors, *22nd International Symposium on Temporal Representation and Reasoning, TIME 2015, Kassel, Germany, September 23-25, 2015*, pages 170–179. IEEE Computer Society, 2015.

[43] Vadim Malvone, Fabio Mogavero, Aniello Murano, and Loredana Sorrentino. Reasoning about graded strategy quantifiers. *Inf. Comput.*, 259(3):390–411, 2018.

[44] Benjamin Aminof, Vadim Malvone, Aniello Murano, and Sasha Rubin. Graded strategy logic: Reasoning about uniqueness of nash equilibria. In Catholijn M. Jonker, Stacy Marsella, John Thangarajah, and Karl Tuyls, editors, *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems, Singapore, May 9-13, 2016*, pages 698–706. ACM, 2016.

[45] Benjamin Aminof, Vadim Malvone, Aniello Murano, and Sasha Rubin. Graded modalities in strategy logic. *Inf. Comput.*, 261:634–649, 2018.

[46] Vadim Malvone and Silvia Stranieri. Towards a model checking tool for strategy logic with simple goals. In Claudio Sacerdoti Coen and Ivano Salvo, editors, *Proceedings of the*

22nd Italian Conference on Theoretical Computer Science, Bologna, Italy, September 13-15, 2021, volume 3072 of *CEUR Workshop Proceedings*, pages 311–316. CEUR-WS.org, 2021.

[47] Davide Catta, Jean Leneutre, and Vadim Malvone. Subset sabotage games & attack graphs. In Angelo Ferrando and Viviana Mascardi, editors, *Proceedings of the 23rd Workshop "From Objects to Agents", Genova, Italy, September 1-3, 2022*, volume 3261 of *CEUR Workshop Proceedings*, pages 209–218. CEUR-WS.org, 2022.

[48] Davide Catta, Jean Leneutre, and Vadim Malvone. Attack graphs & subset sabotage games. *Intelligenza Artificiale*, 17(1):77–88, 2023.

[49] Angelo Ferrando and Vadim Malvone. How to find good coalitions to achieve strategic objectives. In Ana Paula Rocha, Luc Steels, and H. Jaap van den Herik, editors, *Proceedings of the 15th International Conference on Agents and Artificial Intelligence, ICAART 2023, Volume 1, Lisbon, Portugal, February 22-24, 2023*, pages 105–113. SCITEPRESS, 2023.

[50] Davide Catta, Jean Leneutre, and Vadim Malvone. Obstruction logic: A strategic temporal logic to reason about dynamic game models. In Kobi Gal, Ann Nowé, Grzegorz J. Nalepa, Roy Fairstein, and Roxana Radulescu, editors, *ECAI 2023 - 26th European Conference on Artificial Intelligence, September 30 - October 4, 2023, Kraków, Poland - Including 12th Conference on Prestigious Applications of Intelligent Systems (PAIS 2023)*, volume 372 of *Frontiers in Artificial Intelligence and Applications*, pages 365–372. IOS Press, 2023.

[51] Davide Catta, Jean Leneutre, Vadim Malvone, and Aniello Murano. Obstruction alternating-time temporal logic: a strategic logic to reason about dynamic models. In *Proceedings of the 2024 International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2024*, (to appear).

[52] Francesco Belardinelli and Vadim Malvone. A three-valued approach to strategic abilities under imperfect information. In Diego Calvanese, Esra Erdem, and Michael Thielscher, editors, *Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning, KR 2020, Rhodes, Greece, September 12-18, 2020*, pages 89–98, 2020.

[53] Angelo Ferrando and Vadim Malvone. Combine model checking and runtime verification in multi-agent systems. In Claudio Sacerdoti Coen and Ivano Salvo, editors, *Proceedings of the 22nd Italian Conference on Theoretical Computer Science, Bologna, Italy, September 13-15, 2021*, volume 3072 of *CEUR Workshop Proceedings*, pages 302–310. CEUR-WS.org, 2021.

[54] Angelo Ferrando and Vadim Malvone. Strategy RV: A tool to approximate ATL model checking under imperfect information and perfect recall. In Frank Dignum, Alessio Lomuscio, Ulle Endriss, and Ann Nowé, editors, *AAMAS '21: 20th International Conference on Autonomous Agents and Multiagent Systems, Virtual Event, United Kingdom, May 3-7, 2021*, pages 1764–1766. ACM, 2021.

[55] Angelo Ferrando and Vadim Malvone. Give me a hand: How to use model checking for multi-agent systems to help runtime verification and vice versa (short paper). In Riccardo De Benedictis, Nicola Gatti, Marco Maratea, Andrea Micheli, Aniello Murano, Enrico Scala, Luciano Serafini, Ivan Serina, Alessandro Umbrico, and Mauro Vallati, editors, *Proceedings of the 10th Italian workshop on Planning and Scheduling (IPS 2022), RCRA Incontri E Confronti (RiCeRcA 2022), and the workshop on Strategies, Prediction, Interaction, and Reasoning in Italy (SPIRIT 2022) co-located with 21st International Conference of the Italian Association for Artificial Intelligence (AIxIA 2022), November 28 - December 2, 2022, University of Udine, Udine, Italy*, volume 3345 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2022.

[56] Francesco Belardinelli, Angelo Ferrando, Wojciech Jamroga, Vadim Malvone, and Aniello Murano. Scalable verification of strategy logic through three-valued abstraction. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023, 19th-25th August 2023, Macao, SAR, China*, pages 46–54. ijcai.org, 2023.

[57] Solofomampionona Fortunat Rajaona, Ioana Boureanu, Vadim Malvone, and Francesco Belardinelli. Program semantics and verification technique for ai-centred programs. In Marsha Chechik, Joost-Pieter Katoen, and Martin Leucker, editors, *Formal Methods - 25th International Symposium, FM 2023, Lübeck, Germany, March 6-10, 2023, Proceedings*, volume 14000 of *Lecture Notes in Computer Science*, pages 473–491. Springer, 2023.

[58] Francesco Belardinelli, Ioana Boureanu, Vadim Malvone, and Fortunat Rajaona. Automatically verifying expressive epistemic properties of programs. In Brian Williams, Yiling Chen, and Jennifer Neville, editors, *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*, pages 6245–6252. AAAI Press, 2023.

[59] Angelo Ferrando and Vadim Malvone. Runtime verification with imperfect information through indistinguishability relations. In Bernd-Holger Schlingloff and Ming Chai, editors, *Software Engineering and Formal Methods - 20th International Conference, SEFM 2022, Berlin, Germany, September 26-30, 2022, Proceedings*, volume 13550 of *Lecture Notes in Computer Science*, pages 335–351. Springer, 2022.

[60] Francesco Belardinelli and Vadim Malvone. Decidable verification of agent-based data-aware systems. In Matteo Baldoni, Mehdi Dastani, Beishui Liao, Yuko Sakurai, and Rym Zalila-Wenkstern, editors, *PRIMA 2019: Principles and Practice of Multi-Agent Systems - 22nd International Conference, Turin, Italy, October 28-31, 2019, Proceedings*, volume 11873 of *Lecture Notes in Computer Science*, pages 52–68. Springer, 2019.

[61] Wojciech Jamroga and Wiebe van der Hoek. Agents that know how to play. *Fundam. Informaticae*, 63(2-3):185–219, 2004.

[62] Thomas Ågotnes and Dirk Walther. A logic of strategic ability under bounded memory. *J. Log. Lang. Inf.*, 18(1):55–77, 2009.

[63] Natasha Alechina, Brian Logan, Nguyen Hoang Nga, and Abdur Rakib. Logic for coalitions with bounded resources. *J. Log. Comput.*, 21(6):907–937, 2011.

[64] Natasha Alechina, Brian Logan, Nguyen Hoang Nga, and Abdur Rakib. Resource-bounded alternating-time temporal logic. In Wiebe van der Hoek, Gal A. Kaminka, Yves Lespérance, Michael Luck, and Sandip Sen, editors, *9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010), Toronto, Canada, May 10-14, 2010, Volume 1-3*, pages 481–488. IFAAMAS, 2010.

[65] Nils Bulling and Berndt Farwer. Expressing properties of resource-bounded systems: The logics rtl* and RTL. In Jürgen Dix, Michael Fisher, and Peter Novák, editors, *Computational Logic in Multi-Agent Systems - 10th International Workshop, CLIMA X, Hamburg, Germany, September 9-10, 2009, Revised Selected and Invited Papers*, volume 6214 of *Lecture Notes in Computer Science*, pages 22–45. Springer, 2009.

[66] Nils Bulling and Berndt Farwer. On the (un-)decidability of model checking resource-bounded agents. In Helder Coelho, Rudi Studer, and Michael J. Wooldridge, editors, *ECAI 2010 - 19th European Conference on Artificial Intelligence, Lisbon, Portugal, August 16-20, 2010, Proceedings*, volume 215 of *Frontiers in Artificial Intelligence and Applications*, pages 567–572. IOS Press, 2010.

[67] Mehmet Barlo, Guilherme Carmona, and Hamid Sabourian. Bounded memory folk theorem. *J. Econ. Theory*, 163:728–774, 2016.

[68] Anshul Gupta, Sven Schewe, and Dominik Wojtczak. Making the best of limited memory in multi-player discounted sum games. In Javier Esparza and Enrico Tronci, editors, *Proceedings Sixth International Symposium on Games, Automata, Logics and Formal Verification, GandALF 2015, Genoa, Italy, 21-22nd September 2015*, volume 193 of *EPTCS*, pages 16–30, 2015.

[69] Thomas Ågotnes. Action and knowledge in alternating-time temporal logic. *Synth.*, 149(2):375–407, 2006.

[70] Dirk Walther, Wiebe van der Hoek, and Michael J. Wooldridge. Alternating-time temporal logic with explicit strategies. In Dov Samet, editor, *Proceedings of the 11th Conference on Theoretical Aspects of Rationality and Knowledge (TARK-2007), Brussels, Belgium, June 25-27, 2007*, pages 269–278, 2007.

[71] David Harel, Dexter Kozen, and Rohit Parikh. Process logic: Expressiveness, decidability, completeness. *J. Comput. Syst. Sci.*, 25(2):144–170, 1982.

[72] Hein Duijf and Jan M. Broersen. Representing strategies. In Alessio Lomuscio and Moshe Y. Vardi, editors, *Proceedings of the 4th International Workshop on Strategic Reasoning, SR 2016, New York City, USA, 10th July 2016*, volume 218 of *EPTCS*, pages 15–26, 2016.

[73] Rafael H. Bordini, Michael Fisher, Willem Visser, and Michael J. Wooldridge. Verifying multi-agent programs by model checking. *Auton. Agents Multi Agent Syst.*, 12(2):239–256, 2006.

[74] Natasha Alechina, Mehdi Dastani, Brian Logan, and John-Jules Ch. Meyer. A logic of agent programs. In *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence, July 22-26, 2007, Vancouver, British Columbia, Canada*, pages 795–800. AAAI Press, 2007.

[75] Natasha Alechina, Brian Logan, Mehdi Dastani, and John-Jules Ch. Meyer. Reasoning about agent execution strategies. In Lin Padgham, David C. Parkes, Jörg P. Müller, and Simon Parsons, editors, *7th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008), Estoril, Portugal, May 12-16, 2008, Volume 3*, pages 1455–1458. IFAAMAS, 2008.

[76] Nitin Yadav and Sebastian Sardiña. Reasoning about agent programs using atl-like logics. In Luis Fariñas del Cerro, Andreas Herzig, and Jérôme Mengin, editors, *Logics in Artificial Intelligence - 13th European Conference, JELIA 2012, Toulouse, France, September 26-28, 2012. Proceedings*, volume 7519 of *Lecture Notes in Computer Science*, pages 437–449. Springer, 2012.

[77] Steen Vester. Alternating-time temporal logic with finite-memory strategies. In Gabriele Puppis and Tiziano Villa, editors, *Proceedings Fourth International Symposium on Games, Automata, Logics and Formal Verification, GandALF 2013, Borca di Cadore, Dolomites, Italy, 29-31th August 2013*, volume 119 of *EPTCS*, pages 194–207, 2013.

[78] Kerem Kaynar. A taxonomy for attack graph generation and usage in network security. *J. Inf. Secur. Appl.*, 29:27–56, 2016.

[79] Jin-Hee Cho, Dilli P. Sharma, Hooman Alavizadeh, Seunghyun Yoon, Noam Ben-Asher, Terrence J. Moore, Dong Seong Kim, Hyuk Lim, and Frederica Free-Nelson. Toward proactive, adaptive defense: A survey on moving target defense. *IEEE Commun. Surv. Tutorials*, 22(1):709–745, 2020.

[80] Aniello Murano, Giuseppe Perelli, and Sasha Rubin. Multi-agent path planning in known dynamic environments. In Qingliang Chen, Paolo Torroni, Serena Villata, Jane Yung-jen Hsu, and Andrea Omicini, editors, *PRIMA 2015: Principles and Practice of Multi-Agent Systems - 18th International Conference, Bertinoro, Italy, October 26-30, 2015, Proceedings*, volume 9387 of *Lecture Notes in Computer Science*, pages 218–231. Springer, 2015.

[81] Johan van Benthem. An essay on sabotage and obstruction. In Dieter Hutter and Werner Stephan, editors, *Mechanizing Mathematical Reasoning, Essays in Honor of Jörg H. Siekmann on the Occasion of His 60th Birthday*, volume 2605 of *Lecture Notes in Computer Science*, pages 268–276. Springer, 2005.

[82] Christof Löding and Philipp Rohde. Model checking and satisfiability for sabotage modal logic. In Paritosh K. Pandya and Jaikumar Radhakrishnan, editors, *FST TCS 2003: Foundations of Software Technology and Theoretical Computer Science, 23rd Conference, Mumbai, India, December 15-17, 2003, Proceedings*, volume 2914 of *Lecture Notes in Computer Science*, pages 302–313. Springer, 2003.

[83] Guillaume Aucher, Johan van Benthem, and Davide Grossi. Modal logics of sabotage revisited. *J. Log. Comput.*, 28(2):269–303, 2018.

[84] Thomas Ågotnes, Wiebe van der Hoek, Juan A. Rodríguez-Aguilar, Carles Sierra, and Michael J. Wooldridge. On the logic of normative systems. In Manuela M. Veloso, editor, *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007*, pages 1175–1180, 2007.

[85] Orna Kupferman and Moshe Y. Vardi. Module checking. In Rajeev Alur and Thomas A. Henzinger, editors, *Computer Aided Verification, 8th International Conference, CAV '96, New Brunswick, NJ, USA, July 31 - August 3, 1996, Proceedings*, volume 1102 of *Lecture Notes in Computer Science*, pages 75–86. Springer, 1996.

[86] Laura Bozzelli, Aniello Murano, and Adriano Peron. Module checking of pushdown multi-agent systems. In Diego Calvanese, Esra Erdem, and Michael Thielscher, editors, *Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning, KR 2020, Rhodes, Greece, September 12-18, 2020*, pages 162–171, 2020.

[87] Laura Bozzelli and Aniello Murano. On the complexity of ATL and atl* module checking. In Patricia Bouyer, Andrea Orlandini, and Pierluigi San Pietro, editors, *Proceedings Eighth International Symposium on Games, Automata, Logics and Formal Verification, GandALF 2017, Roma, Italy, 20-22 September 2017*, volume 256 of *EPTCS*, pages 268–282, 2017.

[88] Xinming Ou and Anoop Singhal. *Quantitative Security Risk Assessment of Enterprise Networks*. Springer Briefs in Computer Science. Springer, 2011.

[89] Raphaël Berthon, Bastien Maubert, Aniello Murano, Sasha Rubin, and Moshe Y. Vardi. Strategy logic with imperfect information. *ACM Trans. Comput. Log.*, 22(1):5:1–5:51, 2021.

[90] Francesco Belardinelli, Alessio Lomuscio, Aniello Murano, and Sasha Rubin. Verification of multi-agent systems with imperfect information and public actions. In Kate Larson, Michael Winikoff, Sanmay Das, and Edmund H. Durfee, editors, *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2017, São Paulo, Brazil, May 8-12, 2017*, pages 1268–1276. ACM, 2017.

[91] Francesco Belardinelli, Alessio Lomuscio, Aniello Murano, and Sasha Rubin. Verification of broadcasting multi-agent systems against an epistemic strategy logic. In Carles Sierra, editor, *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 91–97. ijcai.org, 2017.

[92] Sharon Shoham and Orna Grumberg. Monotonic abstraction-refinement for CTL. In Kurt Jensen and Andreas Podelski, editors, *Tools and Algorithms for the Construction and Analysis of Systems, 10th International Conference, TACAS 2004, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2004, Barcelona, Spain, March 29 - April 2, 2004, Proceedings*, volume 2988 of *Lecture Notes in Computer Science*, pages 546–560. Springer, 2004.

[93] Sharon Shoham and Orna Grumberg. A game-based framework for CTL counterexamples and 3-valued abstraction-refinement. *ACM Trans. Comput. Log.*, 9(1):1, 2007.

[94] Benjamin Aminof, Orna Kupferman, and Aniello Murano. Improved model checking of hierarchical systems. *Inf. Comput.*, 210:68–86, 2012.

[95] Orna Grumberg, Martin Lange, Martin Leucker, and Sharon Shoham. When not losing is better than winning: Abstraction and refinement for the full mu-calculus. *Inf. Comput.*, 205(8):1130–1148, 2007.

[96] Guy Avni, Shibashis Guha, and Orna Kupferman. An abstraction-refinement methodology for reasoning about network games. In Carles Sierra, editor, *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 70–76. ijcai.org, 2017.

[97] Luca de Alfaro and Pritam Roy. Solving games via three-valued abstraction refinement. *Inf. Comput.*, 208(6):666–676, 2010.

[98] Rayna Dimitrova and Bernd Finkbeiner. Abstraction refinement for games with incomplete information. In Ramesh Hariharan, Madhavan Mukund, and V. Vinay, editors, *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2008, December 9-11, 2008, Bangalore, India*, volume 2 of *LIPIcs*, pages 175–186. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2008.

[99] Mika Cohen, Mads Dam, Alessio Lomuscio, and Francesco Russo. Abstraction in model checking multi-agent systems. In Carles Sierra, Cristiano Castelfranchi, Keith S. Decker, and Jaime Simão Sichman, editors, *8th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2009), Budapest, Hungary, May 10-15, 2009, Volume 2*, pages 945–952. IFAAMAS, 2009.

[100] Francesco Belardinelli and Alessio Lomuscio. A three-value abstraction technique for the verification of epistemic properties in multi-agent systems. In Loizos Michael and Antonis C. Kakas, editors, *Logics in Artificial Intelligence - 15th European Conference, JELIA 2016, Larnaca, Cyprus, November 9-11, 2016, Proceedings*, volume 10021 of *Lecture Notes in Computer Science*, pages 112–126, 2016.

[101] Thomas Ball and Orna Kupferman. An abstraction-refinement framework for multi-agent systems. In *21th IEEE Symposium on Logic in Computer Science (LICS 2006), 12-15 August 2006, Seattle, WA, USA, Proceedings*, pages 379–388. IEEE Computer Society, 2006.

[102] Alessio Lomuscio and Jakub Michaliszyn. An abstraction technique for the verification of multi-agent systems against ATL specifications. In Chitta Baral, Giuseppe De Giacomo, and Thomas Eiter, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the Fourteenth International Conference, KR 2014, Vienna, Austria, July 20-24, 2014*. AAAI Press, 2014.

[103] Alessio Lomuscio and Jakub Michaliszyn. Verification of multi-agent systems via predicate abstraction against ATLK specifications. In Catholijn M. Jonker, Stacy Marsella, John Thangarajah, and Karl Tuyls, editors, *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems, Singapore, May 9-13, 2016*, pages 662–670. ACM, 2016.

[104] Wojciech Jamroga, Beata Konikowska, Damian Kurpiewski, and Wojciech Penczek. Multi-valued verification of strategic ability. *Fundam. Informaticae*, 175(1-4):207–251, 2020.

[105] Angelo Ferrando and Vadim Malvone. VITAMIN: A compositional framework for model checking of multi-agent systems. *CoRR*, abs/2403.02170, 2024.

[106] Gabriel Ballot, Vadim Malvone, Jean Leneutre, and Youssef Laarouchi. Strategic reasoning under capacity-constrained agents. In *Proceedings of the 2024 International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2024*, (to appear).