Verifying Strategic Abilities in Multi-agent Systems via First-order Entailment

Francesco Belardinelli [1] and **Vadim Malvone** [2]

Abstract. The verification of strategic abilities of autonomous agents is a key subject of investigation in the applications of formal methods to the design and certification of multi-agents systems. In this contribution we propose a novel approach to this verification problem. Inspired by recent advances, we introduce a translation from Alternating-time Temporal Logic (ATL) to First-order Logic (FOL). We show that our translation is sound on a fragment of ATL, that we call ATL-live, as it is suitable to express liveness properties in MAS. Further, we show how the universal model checking problem for ATL-live can be reduced to semantic entailment in FOL. Finally, we prove that ATL-live is maximal in the sense that if any other ATL connective is added, non-FOL reasoning techniques would be required. These results are meant to be a first step towards the application of FOL reasoners to model check strategic abilities expressed in ATL.

1 Introduction

The verification of strategic abilities of autonomous agents is a key subject of investigation in the applications of formal methods to the design and certification of multi-agents systems (MAS) [20, 38]. In recent years logic-based languages to model temporal [30, 9], epistemic [13, 28], and strategic [2, 29] capabilities of agents have been introduced and their theoretical properties analysed, particularly in relation with their satisfaction and model checking problems [6, 31]. Results in this area have led to the development of model checking tools [1, 14, 22, 26] that have been successfully applied to MAS verification in domains as diverse as voting protocols [4, 21], robot swarms [24], and AUVs [12]. However, the complexity of multi-agent scenarios means that the verification task is often computationally costly, viz. undecidable [11]; hence the need for decidability results and for more efficient model checking tools and techniques.

Reasoners for First-order Logic (FOL), including SMT solvers, have become the core engine behind a range of powerful technologies and a thriving research area with many practical applications [10]. In the context of temporal logics, [36, 35] present the theoretical foundations for reducing the model checking problem for a fragment of computational-tree logic (CTL), over finite and infinite Kripke structures expressed in FOL, to checking semantic entailment in FOL. In [37] the same authors demonstrate that it is practical to verify temporal properties of infinite-state systems expressed in the same fragment of CTL by using an SMT solver, without iteration, abstraction, or human intervention. Moreover, they show that, by using their method, the verification of a leader election protocol with an unbounded number of processes may in some cases terminate faster than when considering a finite number of processes.

Inspired by these works, in this contribution we propose a novel approach to the verification problem for Alternating-time Temporal Logic (ATL), one of the most well-studied logic-based languages for representing strategic abilities of individuals and coalitions [2, 32]. Specifically, in Sec. 2 after introducing preliminary notions on ATL and FOL, we present a fragment of ATL, that we call ATL-live following [36, 35], as it is suitable to express liveness properties in MAS. In Sec. 3 we present a translation into FOL for concurrent game structures (CGSs), the semantics of choice for ATL. Then, in Sec. 4 we show how the universal model checking problem for ATL-live can be reduced to semantic entailment in FOL. These results are meant to be a first step towards the application of FOL reasoners to model check strategic abilities expressed in ATL. The interesting experimental results discussed in [37] might hint at significant savings in execution time, thus allowing for the verification of more complex multi-agent scenarios. Finally, in Section 5 we prove that ATL-live is maximal in the sense that if any other ATL connective is added, non-FOL reasoning techniques would be required.

Related work. FOL reasoners and SAT/SMT solvers are increasingly applied to the verification of temporal properties of reactive systems [10] (e.g., [3] put forward an SMT-based decision procedure for general modal logic). More specifically, in [25] the authors introduce an explicit reasoning framework for linear temporal logic (LTL), which is built on top of propositional satisfiability (SAT) solving. This approach is then extended in the same reference to reason about *assertional* LTL formulas, where Boolean atoms are replaced with assertions about program variables (e.g., $k \leq 5$), and the underlying SAT solver is replaced by an SMT solver. Further, [15] describes a declarative and deductive symbolic model checker modulo theories (MCMT) for safety properties of infinite state systems. The key component of MCMT is a backward reachability procedure that symbolically computes pre-images of the set of unsafe states and checks for safety and fix-points by solving SMT problems. This framework was extended in [7] to handle complex data-aware business processes with the ability of operating over case and persistent data. Moreover, in [33] a first-order extension of LTL is considered, and a prototype tool based on SMT-based model checking is presented. However, none of the contributions above tackles expressive logic-based languages for reasoning about strategies, such as the logic ATL here considered.

In the context of logics for strategic reasoning, [27] develops a predicate abstraction technique for the verification of multi-agent systems against specifications expressed in an epistemic extension of ATL. In particular, an infinite-state multi-agent program is reduced to a finite model by predicate abstraction, where predicates are gener-

[1] Imperial College London, United Kingdom and Laboratoire IBISC, Universite dEvry, France
[2] Laboratoire IBISC, Universite dEvry, France

ated automatically via SMT calls. In a related field, [16] puts forward an SMT-based approach to verifying purely epistemic properties of programs.

To conclude, to the best of our knowledge, this is the first contribution that tackles the model checking problem for ATL via first-order reasoning.

2 Background

In this section we introduce standard terminology and notation on First-order Logics (Sec. 2.1) and Alternating-time Temporal Logic (Sec. 2.2). We also introduce ATL-live, a fragment of ATL expressive enough to represent reachability goals.

2.1 First order logic

Formulas in First-order Logic (FOL) are built from individual variables and function and relation symbols, by using logical connectives [19]. The set of logical connectives and their meaning in FOL is fixed. The following is a standard set of logical connectives for FOL: the Boolean connectives negation $\neg$ and implication $\rightarrow$, the universal quantifier $\forall$, and equality $=$. On the other hand, the set of function and relation symbols, as well as their semantics, depends on the context. Since for different applications different sets of function and relation symbols are used, we consider the following:

Definition 1 (Base) *A* base *for FOL is a pair $\mathcal{B} = \langle \mathcal{F}, \mathcal{R} \rangle$ such that $\mathcal{F}$ is a set of function symbols and $\mathcal{R}$ is a set of relation symbols.*

Every function and relation symbol has a corresponding arity. Constants are function symbols with arity 0. A (function or relation) symbol X with arity m, for $m \geq 0$, is denoted by X/m.

We now recall the syntax of first-order logic (including equality). In the rest of the paper we fix a set *Var* of (individual) variables.

Definition 2 (FO-formulas) *Given a base $\mathcal{B}$, the formulas φ in the first-order language $\mathcal{L}_{\mathcal{B}}$ are defined by the following BNF:*

$$\begin{array}{rcl} t & ::= & x \mid f(t_1, \ldots, t_m) \\ \varphi & ::= & P(t_1, \ldots, t_m) \mid t = t' \mid \neg\varphi \mid \varphi \rightarrow \varphi \mid \forall x \varphi \end{array}$$

where $x \in Var$, $f/m \in \mathcal{F}$, $P/m \in \mathcal{R}$, $t_1, \ldots, t_m$ is a m-tuple of terms, and t, t' are terms.

The constants $\top$ and $\bot$, disjunction $\vee$, conjunction $\wedge$, coimplication $\leftrightarrow$, and the existential quantifier $\exists$ can be introduced as standard.

The semantics of FOL formulas is defined by using interpretations. An interpretation fixes the meaning of a base by assigning values to variables, function and relation symbols.

Definition 3 (Interpretation) *Given a base $\mathcal{B} = \langle \mathcal{F}, \mathcal{R} \rangle$, an* interpretation *is a pair $\mathcal{I} = \langle D, \cdot^{\mathcal{I}} \rangle$, where D is a non-empty set, the* domain *of $\mathcal{I}$, and $\cdot^{\mathcal{I}}$ is a mapping that assigns:*

1. *to every function symbol $f/m \in \mathcal{F}$ of arity $m \geq 0$, a total m-ary function $f^{\mathcal{I}}$ from D^m to D;*
2. *to every relation symbol $R/m \in \mathcal{R}$, a subset $R^{\mathcal{I}} \subseteq D^m$.*

By Def. 3, the case of 0-ary symbols is dealt with as follows:

- every 0-ary $c \in \mathcal{F}$ is assigned an element $c^{\mathcal{I}} \in D$;
- every 0-ary $p \in \mathcal{R}$ is assigned either true or false.

To fix the meaning of individual variables, we introduce the notion of assignment as a function σ from variables in *Var* to elements in D. We denote by σ^x_u the assignment such that *(i)* $\sigma^x_u(x) = u$; and *(ii)* $\sigma^x_u(x') = \sigma(x')$ for every $x' \neq x$.

Given an interpretation $\mathcal{I}$ and assignment σ, we can specify the interpretation $t^{\mathcal{I}}$ of a term t inductively as follows:

- for $t = x$, $t^{\mathcal{I}} = \sigma(x)$
- for $t = f(t_1, \ldots, t_m)$, $t^{\mathcal{I}} = f^{\mathcal{I}}(t_1^{\mathcal{I}}, \ldots, t_m^{\mathcal{I}})$

Finally, we present the semantics of First-order logic.

Definition 4 (Satisfaction of FOL formulas) *Given a base $\mathcal{B} = \langle \mathcal{F}, \mathcal{R} \rangle$, an interpretation $\mathcal{I} = \langle D, \cdot^{\mathcal{I}} \rangle$, and an FO-formula $\varphi \in \mathcal{L}_{\mathcal{B}}$, we inductively define whether $\mathcal{I}$ satisfies φ under assignment σ, or $(\mathcal{I}, \sigma) \models \varphi$, as follows:*

$$\begin{array}{lll} (\mathcal{I}, \sigma) \models R(t_1, \ldots, t_m) & \textit{iff} & (t_1^{\mathcal{I}}, \ldots, t_m^{\mathcal{I}}) \in R^{\mathcal{I}} \\ (\mathcal{I}, \sigma) \models t_1 = t_2 & \textit{iff} & t_1^{\mathcal{I}} = t_2^{\mathcal{I}} \\ (\mathcal{I}, \sigma) \models \neg\varphi & \textit{iff} & (\mathcal{I}, \sigma) \not\models \varphi \\ (\mathcal{I}, \sigma) \models \varphi \rightarrow \varphi' & \textit{iff} & (\mathcal{I}, \sigma) \not\models \varphi \textit{ or } (\mathcal{I}, \sigma) \models \varphi' \\ (\mathcal{I}, \sigma) \models \forall x \varphi & \textit{iff} & \textit{for all } u \in D,\ (\mathcal{I}, \sigma^x_u) \models \varphi \end{array}$$

We say that a formula φ is *true* in $\mathcal{I}$, or $\mathcal{I} \models \varphi$, iff $(\mathcal{I}, \sigma) \models \varphi$ for all assignments σ, or equivalently, $\mathcal{I}$ is a *model* of φ.

We now present the standard notion of semantic entailment in FOL, which plays a key role in the rest of the paper.

Definition 5 (Entailment) *Let Γ be a set of FOL formulas and φ an FOL formula.* Γ entails φ, *or* $\Gamma \models \varphi$, *iff every interpretation that satisfies all the formulas in Γ also satisfies φ:*

$$\Gamma \models \varphi \quad \textit{iff} \quad \textit{for every } \mathcal{I},\ \textit{if } \mathcal{I} \models \psi \textit{ for every } \psi \in \Gamma,\ \textit{then } \mathcal{I} \models \varphi$$

Semantic entailment checking for FOL is recursively enumerable [8]. This means that semantic entailment checking for FOL is not computable in general, but there is procedure that given Γ and φ produces a proof in the case where $\Gamma \models \varphi$. However, many first-order theories of interest, such as the real and rational numbers, and the monadic and guarded fragments, have been proved decidable [17, 18].

2.2 Alternating-time Temporal Logic and ATL-live

In this section we recall syntax and semantics of Alternating-time Temporal Logic ATL and present the ATL-live fragment.

To fix the notation, we assume that *Ag* is the set of agents and AP the set of atomic propositions. We denote the length of a tuple u as $|u|$ and its i-th element as u_i. For $i \leq |u|$, let $u_{\geq i}$ be the suffix $u_i, \ldots, u_{|u|}$ of u starting at u_i and $u_{\leq i}$ its prefix $u_1, \ldots, u_i$.

Syntax. To reason about the strategic abilities of agents, we use the Alternating-time Temporal Logic ATL [2].

Definition 6 (ATL) *Formulas in ATL are defined as follows, where $q \in AP$ and $A \subseteq Ag$:*

$$\varphi \quad ::= \quad q \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle\langle A \rangle\rangle X\varphi \mid \langle\langle A \rangle\rangle \varphi U \varphi$$

As customary, a formula $\langle\langle A \rangle\rangle \phi$ is read as "the agents in coalition A have a strategy to achieve ϕ". The meaning of linear-time operators *next* X and *until* U is standard [23]. Operators $[\![A]\!]$, *release* R, *finally* F, and *globally* G can be introduced as usual. For instance, $\langle\langle A \rangle\rangle F\phi \equiv \langle\langle A \rangle\rangle \top U \phi$ and $[\![A]\!] G\phi \equiv \neg(\langle\langle A \rangle\rangle F \neg\phi)$.

Inspired by the relationship between the CTL and CTL-live introduced in [35], we introduce a novel fragment of ATL that we call ATL-live, as it contains the ATL connectives that are normally used to express liveness properties. In particular, ATL-live can be model checked directly using an FOL reasoner. Formally, ATL-live is defined as follows.

Definition 7 (ATL-live) *Temporal (φ) and propositional (ϕ) formulas in ATL-live are defined as follows, where $q \in AP$ and $A \subseteq Ag$:*

$$\begin{array}{lcl} \varphi & ::= & \phi \,|\, \varphi \vee \varphi \,|\, \varphi \wedge \varphi \,|\, \langle\!\langle A \rangle\!\rangle X\varphi \,|\, [\![A]\!] X\varphi \,|\, \langle\!\langle A \rangle\!\rangle \varphi U \varphi \,|\, [\![A]\!] \varphi U \varphi \\ \phi & ::= & q \mid \neg\phi \mid \phi \wedge \phi \end{array}$$

Formulas in ATL-live are all and only the temporal formulas.

ATL-live disallows a temporal connective to be in the scope of negation $\neg$, which can only be applied to propositional formulas. E.g., the ATL formula $\neg(\langle\!\langle A \rangle\!\rangle p U q)$ is not allowed, but $\langle\!\langle A \rangle\!\rangle (\neg p) U q$ is. Equivalently ATL-live can be seen as the fragment of the Alternating Modal μ-calculus restricted to the μ operator (where negation is only applied to atoms).

Semantics. To interpret formulas in ATL and ATL-live we make use of game-like structures.

Definition 8 (CGS) *A* concurrent game structure (CGS) *is a tuple $\mathcal{G} = \langle Ag, S, s_0, \{Act_a\}_{a \in Ag}, \tau, L\rangle$ such that*

- *Ag is a set of agents;*
- *S is a non-empty set of* states *and $s_0 \in S$ is the* initial state*;*
- *for every agent $a \in Ag$, Act_a is a non-empty set of* actions*, and $ACT = \prod_{a \in Ag} Act_a$ is the set of* joint actions*;*
- *$\tau : S \times ACT \to S$ is the* transition function*;*
- *$L : S \to 2^{AP}$ is the* labelling function*.*

A *path* is a (finite or infinite) sequence $\pi \in S^* \cup S^\omega$ such that for every $j \geq 1$, $\pi_{j+1} = \tau(\pi_j, \vec{\alpha}_j)$ for some joint action $\vec{\alpha}_j \in ACT$. We distinguish between finite paths, or *histories*, and infinite paths, or *computations*.

When giving a semantics to ATL formulas we assume that agents are endowed with *strategies*.

Definition 9 (Perfect Recall Strategy) *A* strategy with perfect recall *for agent $a \in Ag$ is a function $f_a : S^+ \to Act_a$.*

By Def. 9 any strategy for agent a has to return actions that are enabled for a. Then, given a joint strategy $F_A = \{f_a \mid a \in A\}$, comprising of one strategy for each agent in coalition A, a path p is *F_A-compatible* iff for every $j \geq 1$, $p_{j+1} = \tau(p_j, \vec{\alpha})$ for some joint action $\vec{\alpha}$ such that for every $a \in A$, $\alpha_a = f_a(p_{\leq j})$. Let $out(s, F_A)$ be the set of all such F_A-compatible paths from s.

We can now assign a meaning to ATL (including ATL-live) formulas on CGS.

Definition 10 (Satisfaction) *The satisfaction relation $\models$ for a CGS $\mathcal{G}$, state $s \in S$, path $p \in S^\omega$, atom $q \in AP$, and ATL formula ϕ is defined as follows:*

$$\begin{array}{ll} (\mathcal{G}, s) \models q & \textit{iff } q \in L(s) \\ (\mathcal{G}, s) \models \neg\varphi & \textit{iff } (\mathcal{G}, s) \not\models \varphi \\ (\mathcal{G}, s) \models \varphi \wedge \varphi' & \textit{iff } (\mathcal{G}, s) \models \varphi \textit{ and } (\mathcal{G}, s) \models \varphi' \\ (\mathcal{G}, s) \models \langle\!\langle A \rangle\!\rangle X\varphi & \textit{iff for some } F_A, \textit{for all } p \in out(s, F_A), \\ & \qquad (\mathcal{G}, p_2) \models \varphi \\ (\mathcal{G}, s) \models \langle\!\langle A \rangle\!\rangle \varphi U \varphi' & \textit{iff for some strategy } F_A, \\ & \textit{for all paths } p \in out(s, F_A), \\ & \textit{for some } k \geq 1, (\mathcal{G}, p_k) \models \varphi', \textit{ and} \\ & \textit{for all } j,\ 1 \leq j < k \textit{ implies } (\mathcal{G}, p_j) \models \varphi \end{array}$$

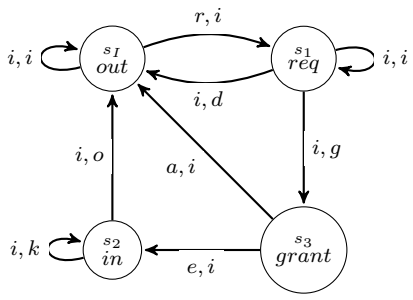

Figure 1. The CGS G for the Train Gate Controller scenario in Example 1.

A formula φ is *true* in a CGS $\mathcal{G}$, or $\mathcal{G} \models \varphi$, iff $(\mathcal{G}, s_0) \models \varphi$. Further, the set of states of a CGS $\mathcal{G}$ that satisfies an ATL formula φ is denoted by $[\varphi]_\mathcal{G} = \{s \in S \mid (\mathcal{G}, s) \models \varphi\}$. We omit $\mathcal{G}$ when clear from the context.

We now state the model checking problem within the present setting.

Definition 11 (Model Checking) *Given a CGS $\mathcal{G}$ and a formula ϕ, the model checking problem amounts to determine whether $\mathcal{G} \models \phi$.*

Definition 12 (Substructure) *A CGS $\mathcal{G}^1 = \langle Ag^1, S^1, s_0^1, \{Act_a^1\}_{a \in Ag^1}, \tau^1, L^1\rangle$ is a substructure of $\mathcal{G}^2 = \langle Ag^2, S^2, s_0^2, \{Act_a^2\}_{a \in Ag^2}, \tau^2, L^2\rangle$, denoted by $\mathcal{G}^1 \sqsubseteq \mathcal{G}^2$, iff $Ag^1 = Ag^2$, $S^1 = S^2$, $s_0^1 = s_0^2$, $\{Act_a^1 = Act_a^2\}_{a \in Ag^1}$, $\tau^1 = \tau^2$, $L^1 \subseteq L^2$.* [3]

By Def. 10 and 12, we immediately obtain the following result.

Proposition 1 *Suppose $\mathcal{G}^1 \sqsubseteq \mathcal{G}^2$ and φ is an ATL formula over L^1, we have that if $\mathcal{G}^1 \models \varphi$ then $\mathcal{G}^2 \models \varphi$.*

To conclude this section we present a toy example that illustrates the formal machinery introduced thus far.

Example 1 *The CGS G depicted in Fig. 1 describes the Train Gate Controller scenario [2]. A train t is outside a gate and it can choose to either stay outside the gate (move i), in s_I, or request (move r) to enter the gate and proceed to s_1. At s_1, the controller c can choose to either grant (move g) the train permission to enter the gate, or deny (move d) the trains request, or delay (move i) the handling of the request. At s_3, the train can choose to either enter the gate (move e) or abandon (move a) its permission to enter the gate. At s_2, the controller can choose to either keep the gate closed (move k) or reopen (move o) the gate to new requests.*

More formally, the CGS G is comprised of the agents in $Ag = \{t, c\}$, atoms in $AP = \{out, req, grant, in\}$, states in $S = \{s_I, s_1, s_2, s_3\}$ with initial state s_I, actions in $Ac_t = \{r, e, a, i\}$ and $Ac_c = \{g, d, k, o, i\}$. Transitions are given as in Fig. 1.

As an example of specifications in ATL, consider the formula $\varphi = \langle\!\langle t \rangle\!\rangle Fin$. This formula can be read as: the train t has a strategy such

[3] By $L^1 \subseteq L^2$ we mean that L^1 is the restriction of L^2 on some subset A of the set AP of atomic propositions.

that sooner or later it enters in the gate. Notice that φ is false in G because, without the grant of controller, t can not be certain to reach state s_2. Another example of specification can be $\varphi = \langle\langle\emptyset\rangle\rangle G(out \rightarrow \langle\langle t, c\rangle\rangle Fin)$, that is whenever the train is out of the gate, the train and the controller can cooperate so that the train will enter the gate.

3 Translating CGS into FOL Theories

In this section we introduce a translation from concurrent game structure to FOL theories, starting with the underlying base.

Definition 13 *Given sets Ag of agents and AP of atoms, the base $\mathcal{B} = \langle \mathcal{F}, \mathcal{R} \rangle$ is such that*

- *$\mathcal{F}$ is empty;*
- *$\mathcal{R} = \{Ag_{\mathcal{B}}/1, S_{\mathcal{B}}/1, S_{0\mathcal{B}}/1, T_{\mathcal{B}}/(|Ag|+2)\} \cup \{Q_{\mathcal{B}}/1 \mid q \in AP\} \cup \{Act_{a\mathcal{B}}/1 \mid a \in Ag\}$*

We omit subscript $\mathcal{B}$ whenever it is clear from the context.

We use the same notation for elements in CGS and $\mathcal{B}$ the distinction will be clear from the context.

First-order formulas over the base in Def. 13 can be used to represent the various components of a CGS: the state space, the initial states, the set of actions for each agent, the transition function, and the labelling function. Then, suitable interpretations satisfying the FOL formulas on $\mathcal{B}$ represent a CGS. Observe that the relation symbols themselves do not represent a CGS, but an interpretation satisfying suitable FOL formulas determines the content of these relation symbols, and therefore represents a CGS. Then, the set of all the satisfying interpretations forms a class of CGSs. We call a set of formulas that represent a class of CGSs a declarative model [34].

Definition 14 (Declarative model) *A* declarative model *is a pair $\mathcal{D} = \langle \mathcal{B}, \Gamma \rangle$, where $\mathcal{B}$ is a base according to Def. 13, and Γ is a set of FOL formulas over $\mathcal{B}$ that includes well-formedness constraints on CGS (e.g., "the set of states is not empty").*

More precisely, Γ includes the following formulas[4]*:*

1. $\exists x S_0(x)$
2. $\forall x(S_0(x) \rightarrow S(x))$
3. $\forall x, x'(S_0(x) \wedge S_0(x') \rightarrow x = x')$
4. $\bigwedge_{a \in Ag} \exists x Act_a(x)$
5. $\forall x \forall y_1, \ldots, y_{|Ag|}(S(x) \wedge Act_1(y_1) \wedge \ldots \wedge Act_{|Ag|}(y_{|Ag|}) \rightarrow \exists x'(S(x') \wedge T(x, y_1, \ldots, y_{|Ag|}, x')))$
6. $\forall x \forall y_1, \ldots, y_{|Ag|} \forall x', x''((S(x) \wedge Act_1(y_1) \wedge \ldots \wedge Act_{|Ag|}(y_{|Ag|}) \wedge S(x') \wedge S(x'') \wedge T(x, y_1, \ldots, y_{|Ag|}, x') \wedge T(x, y_1, \ldots, y_{|Ag|}, x')) \rightarrow (x' = x''))$

By Def. 14 we immediately obtain the following result.

Lemma 2 *Every interpretation $\mathcal{I}$ of a declarative model $\mathcal{D} = \langle \mathcal{B}, \Gamma \rangle$ is a CGS $\mathcal{G}_{\mathcal{I}} = \langle Ag_{\mathcal{I}}, S_{\mathcal{I}}, S_{0\mathcal{I}}, \{Act_{a\,\mathcal{I}}\}_{a \in Ag}, \tau_{\mathcal{I}}, L_{\mathcal{I}} \rangle$ where*

- *$Ag_{\mathcal{I}} = \{u \in D_{\mathcal{I}} \mid u \in Ag_{\mathcal{B}}^{\mathcal{I}}\}$;*
- *$S_{\mathcal{I}} = \{u \in D_{\mathcal{I}} \mid u \in S_{\mathcal{B}}^{\mathcal{I}}\}$;*
- *$S_{0\mathcal{I}} = \{u \in D_{\mathcal{I}} \mid u \in S_{0\mathcal{B}}^{\mathcal{I}}\}$;*
- *for every $a \in Ag_{\mathcal{I}}$, $Act_{a\,\mathcal{I}} = \{u \in D_{\mathcal{I}} \mid u \in Act_{a\mathcal{B}}^{\mathcal{I}}\}$;*
- *for $s, s' \in S_{\mathcal{I}}$, $\alpha \in ACT_{\mathcal{I}}$, $s' = \tau_{\mathcal{I}}(s, \alpha)$ iff $s \in S_{\mathcal{B}}^{\mathcal{I}}$, $s' \in S_{\mathcal{B}}^{\mathcal{I}}$, for every $a \in Ag_{\mathcal{B}}^{\mathcal{I}}$, $\alpha_a \in Act_{a\,\mathcal{I}}$, and $(s, \alpha, s') \in T_{\mathcal{B}}^{\mathcal{I}}$;*
- *for $q \in AP$, $s \in S_{\mathcal{I}}$, $q \in L_{\mathcal{I}}(s)$ iff $s \in Q_{\mathcal{B}}^{\mathcal{I}}$.*

[4] We do not explicitly assume that the various sets composing a CGS are disjoint, as our results do not depend on this assumption.

In particular, since interpretation $\mathcal{I}$ satisfies Γ, the CGS $\mathcal{G}_{\mathcal{I}}$ is well-defined, i.e., it satisfies Def. 8.

The Class of CGSs $\mathcal{G}_{\mathcal{I}}$ represented by some interpretation $\mathcal{I}$ of a declarative model $\mathcal{D} = \langle \mathcal{B}, \Gamma \rangle$ is denoted by

$$CK(\mathcal{D}) \quad = \quad \{\mathcal{G}_{\mathcal{I}} \mid \text{ for all } \psi \in \Gamma, \mathcal{I} \models \psi\}$$

Inclusion of the well-formedness formulas in Γ insures that every member of $CK(\mathcal{D})$ is a valid CGS. There are many reasons for a set of FOL formulae to have more than one satisfying interpretation: the use of uninterpreted functions (relations) can result in more than one satisfying interpretation. Moreover, under-constraining a model makes it possible to have non-isomorphic CGSs that are satisfying interpretations.

By Def. 14 two model checking problems can be studied.

Definition 15 (Universal and Existential Model Checking Problem) *The universal (resp. existential) model checking problem for a declarative model $\mathcal{D}$ and an ATL formula φ, denoted by $\mathcal{D} \models_\forall \varphi$ (resp. $\mathcal{D} \models_\exists \varphi$), is defined as checking whether all (resp. some) CGSs in $CK(\mathcal{D})$ satisfy φ:*

$$\mathcal{D} \models_\forall \varphi \quad \textit{iff} \quad \textit{for all } \mathcal{G} \in CK(\mathcal{D}), \mathcal{G} \models \varphi$$
$$\mathcal{D} \models_\exists \varphi \quad \textit{iff} \quad \textit{for some } \mathcal{G} \in CK(\mathcal{D}), \mathcal{G} \models \varphi$$

Example 2 *An example of declarative model $\mathcal{D} = \langle \mathcal{B}, \Gamma \rangle$ can have:*

- *Base $\mathcal{B} = \langle \mathcal{F}, \mathcal{R} \rangle$:*
  - *$\mathcal{F}$ is empty;*
  - *$\mathcal{R} = \{Ag/1, S/1, S_0/1, Act_t/1, Act_c/1, T/4, Out/1, Req/1, Grant/1, In/1\}$.*
- *Constraints Γ includes formulas (1)-(6) in Def. 14 as well as:*
  - *$Ag(t) \wedge Ag(c)$*
  - *$S(s_I) \wedge S(s_1) \wedge S(s_2) \wedge S(s_3)$*
  - *$S_0(s_I)$*
  - *$Act_t(r) \wedge Act_t(e) \wedge Act_t(a) \wedge Act_t(i)$*
  - *$Act_c(g) \wedge Act_c(d) \wedge Act_c(k) \wedge Act_c(o) \wedge Act_c(i)$*
  - *$T(s_I, i, i, s_I) \wedge T(s_I, r, i, s_1) \wedge T(s_1, i, i, s_1) \wedge T(s_1, i, d, s_I) \wedge T(s_1, i, g, s_3) \wedge T(s_3, a, i, s_I) \wedge T(s_3, e, i, s_2) \wedge T(s_2, i, o, s_I) \wedge T(s_2, i, k, s_2)$*
  - *$Out(s_I) \wedge Req(s_1) \wedge Grant(s_2) \wedge In(s_3)$*

Given $\mathcal{B}$ and Γ as defined above, we have the declarative model $\mathcal{D} = \langle \mathcal{B}, \Gamma \rangle$ in which one of its interpretations is the CGS $\mathcal{G}$ described in Example 1.

4 Model Checking ATL-live

In this section we tackle universal model checking for ATL-live formulas. We also show how this approach can be applied to existential model checking by analysing the relation between these two problems.

To model check a declarative model $\mathcal{D} = \langle \langle \mathcal{F}, \mathcal{R} \rangle, \Gamma \rangle$ and an ATL-live formula φ, we make use of functions *theory* and *axiom* to create an enriched version of $\mathcal{D}$. More in detail, for each subformula ψ of φ, we add a new (characteristic) predicate P_ψ to $\mathcal{R}$, and to Γ the set of constraints related to ψ, as follows.

Definition 16 (Theory) *Given a declarative model $\mathcal{D}$ and an ATL-live formula φ, theory($\mathcal{D}, \varphi$) is inductively defined as follows.*

Case φ of

$$
\begin{array}{ll}
q & \Rightarrow \mathcal{D} \\
\boxtimes\psi & \Rightarrow \langle\langle \mathcal{F}, \mathcal{R} \cup \{P_\varphi/1\}\rangle, \Gamma \cup \mathit{axiom}(\varphi)\rangle \\
 & \quad \textit{where } \langle\langle \mathcal{F}, \mathcal{R}\rangle, \Gamma\rangle = \mathit{theory}(\mathcal{D}, \psi) \\
\psi_1 \otimes \psi_2 & \Rightarrow \langle\langle \mathcal{F}, \mathcal{R}_1 \cup \mathcal{R}_2 \cup \{P_\varphi/1\}\}\rangle, \Gamma_1 \cup \Gamma_2 \cup \mathit{axiom}(\varphi)\rangle \\
 & \quad \textit{where } \langle\langle \mathcal{F}, \mathcal{R}_1\rangle, \Gamma_1\rangle = \mathit{theory}(\mathcal{D}, \psi_1) \\
 & \quad \textit{and } \langle\langle \mathcal{F}, \mathcal{R}_2\rangle, \Gamma_2\rangle = \mathit{theory}(\mathcal{D}, \psi_2)
\end{array}
$$

where $\boxtimes \in \{\neg, \langle\langle\Gamma\rangle\rangle X, [\![\Gamma]\!] X\}$ and $\otimes \in \{\vee, \wedge, \langle\langle\Gamma\rangle\rangle U, [\![\Gamma]\!] U\}$.

Definition 17 (Axiom) *Given an ATL-live formula φ, the set axiom(φ) of FOL formulas is defined as follows.*

Case φ of

$$
\begin{array}{ll}
q & \Rightarrow \forall\, s(P_\varphi(s) \leftrightarrow Q(s)) \\
\neg\phi & \Rightarrow \forall s(P_\varphi(s) \leftrightarrow \neg P_\phi(s)) \\
\phi_1 \wedge \phi_2 & \Rightarrow \forall s(P_\varphi(s) \leftrightarrow (P_{\phi_1}(s) \wedge P_{\phi_2}(s))) \\
\varphi_1 \vee \varphi_2 & \Rightarrow \forall s(P_\varphi(s) \leftrightarrow (P_{\varphi_1}(s) \vee P_{\varphi_2}(s))) \\
\varphi_1 \wedge \varphi_2 & \Rightarrow \forall s(P_\varphi(s) \leftrightarrow (P_{\varphi_1}(s) \wedge P_{\varphi_2}(s))) \\
\langle\langle\Gamma\rangle\rangle X\varphi' & \Rightarrow \forall s(P_\varphi(s) \leftrightarrow \exists s', \alpha_1, \ldots, \alpha_{|\Gamma|} \forall \alpha_{|\Gamma|+1}, \ldots, \alpha_{|Ag\setminus\Gamma|} \\
 & \quad (T(s, \alpha_1, \ldots, \alpha_{|Ag|}, s') \wedge P_{\varphi'}(s'))) \\
\langle\langle\Gamma\rangle\rangle \varphi_1 U \varphi_2 & \Rightarrow \textit{(i)}\ \forall s(P_{\varphi_2}(s) \rightarrow P_\varphi(s))\textit{; (ii)}\ \forall s((P_{\varphi_1}(s) \wedge \\
 & \quad \exists s', \alpha_1, \ldots, \alpha_{|\Gamma|} \forall \alpha_{|\Gamma|+1}, \ldots, \alpha_{|Ag\setminus\Gamma|} \\
 & \quad (T(s, \alpha_1, \ldots, \alpha_{|Ag|}, s') \wedge P_\varphi(s'))) \rightarrow P_\varphi(s)) \\
[\![\Gamma]\!] X\varphi' & \Rightarrow \forall s(P_\varphi(s) \leftrightarrow \forall s', \alpha_1, \ldots, \alpha_{|\Gamma|} \exists \alpha_{|\Gamma|+1}, \ldots, \alpha_{|Ag\setminus\Gamma|} \\
 & \quad (T(s, \alpha_1, \ldots, \alpha_{|Ag|}, s') \rightarrow P_{\varphi'}(s'))) \\
[\![\Gamma]\!] \varphi_1 U \varphi_2 & \Rightarrow \textit{(i)}\ \forall s(P_{\varphi_2}(s) \rightarrow P_\varphi(s))\textit{; (ii)}\ \forall s((P_{\varphi_1}(s) \wedge \\
 & \quad \forall s', \alpha_1, \ldots, \alpha_{|\Gamma|} \exists \alpha_{|\Gamma|+1}, \ldots, \alpha_{|Ag\setminus\Gamma|} \\
 & \quad (T(s, \alpha_1, \ldots, \alpha_{|Ag|}, s') \rightarrow P_\varphi(s'))) \rightarrow P_\varphi(s))
\end{array}
$$

The function *theory* is recursive in the structure of φ. For each operator in ATL-live, the constraints that are added to model $\mathcal{D}$ by *theory* are defined by the (non-recursive) function *axiom*. For every subformula φ' of φ, *axiom* introduces one or two FOL formulas for each new predicate $P_{\varphi'}$, which are then added to $\mathcal{D}$. Observe that the complexity of *theory* is linear in the size of φ.

Example 3 *Given the declarative model of Example 2, $\mathcal{D} = \langle\langle \mathcal{F}, \mathcal{R}\rangle, \Gamma\rangle$ and formula $\varphi = \langle\langle t\rangle\rangle Fin$ analysed in Example 1, we construct a new declarative model $\mathcal{D}' = \langle\langle \mathcal{F}, \mathcal{R}'\rangle, \Gamma'\rangle$, where $\mathcal{R}' = \mathcal{R} \cup P_\varphi$ and Γ' is the union of Γ with the following constraints:*

1. $\forall s(S(s) \wedge P_{in}(s)) \rightarrow P_\varphi(s)$
2. $\forall s(S(s) \wedge \exists s', \alpha_t, \forall \alpha_c(S(s') \wedge Act_t(\alpha_t) \wedge Act_c(\alpha_c) \wedge T(s, \alpha_t, \alpha_c, s') \wedge P_\varphi(s'))) \rightarrow P_\varphi(s)$.

Intuitively, (1) states that every state that satisfies in, also satisfies $\langle\langle t\rangle\rangle Fin$. While, (2) states that if from state s there exists an action for t such that for all the actions of c the resulting state s' satisfies $\langle\langle t\rangle\rangle Fin$, then s also satisfies $\langle\langle t\rangle\rangle Fin$.

Recall that a declarative model is basically a set of constraints that captures CGSs and its models define a class of CGSs, as shown in Lemma 2. Given a declarative model $\mathcal{D}$, every $\mathcal{G} \in CK(\mathcal{D})$ can be seen both as a CGS and as an interpretation of $\mathcal{D}$. As a consequence, both $[\varphi]_\mathcal{G}$ and $P^\mathcal{G}_\varphi$ are sets of states: the extension of $[\varphi]_\mathcal{G}$ is determined by the semantics of ATL and considering $\mathcal{G}$ as a CGS, whereas the extension of $P^\mathcal{G}_\varphi$ is determined by the semantics of FOL and considering $\mathcal{G}$ as a model of $\mathcal{D}$. Hereafter, we explore the properties of the declarative model generated by the function *theory*.

First, we analyse the relationship between the class $CK(\mathcal{D})$ of CGSs defined by the declarative model $\mathcal{D}$ and *theory*($\mathcal{D}, \varphi$). The declarative model *theory*($\mathcal{D}, \varphi$) contains a labelling predicate $P_{\varphi'}$ and some constraints for every subformula φ' of φ. As a result, every CGS in $CK(\mathit{theory}(\mathcal{D}, \varphi))$ can be converted into a CGS in $CK(\mathcal{D})$ by simply dropping the extra labelling predicates, as stated in the following lemma, whose proof is immediate.

Lemma 3 *Let $\mathcal{D}$ be a declarative model and φ an ATL-live formula, for every $\mathcal{G}$ in $CK(\mathit{theory}(\mathcal{D}, \varphi))$ there exists a $\mathcal{G}'$ in $CK(\mathcal{D})$ that is a substructure of $\mathcal{G}$, i.e., $\mathcal{G}' \sqsubseteq \mathcal{G}$.*

Next we investigate the relationship between the set $[\varphi]$ of states that satisfy an ATL-live formula φ and the set of states defined by the labelling predicate P_φ. If ϕ is a propositional formula, as defined in Def. 7, then for every $\mathcal{G}$ in $CK(\mathit{theory}(\mathcal{D}, \phi))$, the sets $[\phi]_\mathcal{G}$ and $P^\mathcal{G}_\phi$ are equal. This is due to the fact that the constraints that are defined in Def. 17 for these connectives are necessary and sufficient to characterize the set of states that satisfy ϕ.

Lemma 4 *Let $\mathcal{D}$ be a declarative model and ϕ a propositional formula as per Def. 7. Then,*

$$\textit{for all } \mathcal{G} \in CK(\mathit{theory}(\mathcal{D}, \phi)),\ [\phi]_\mathcal{G} = P^\mathcal{G}_\phi$$

Proof. The proof is by induction on the structure of ϕ. In the following cases, we assume that $\mathcal{G} \in CK(\mathit{theory}(\mathcal{D}, \phi))$.
Base case: suppose $\phi = q$, for $q \in AP$. By Def. 10 and 17, for every state s, $s \in [q]_\mathcal{G}$ iff $q \in L(s)$, iff $s \in Q^\mathcal{G}$, iff $s \in P^\mathcal{G}_q$. Therefore, $[q]_\mathcal{G} = P^\mathcal{G}_q$.
Induction step: according to the structure of ϕ, two cases are distinguished, with $[\phi_1]_\mathcal{G} = P^\mathcal{G}_{\phi_1}$ and $[\phi_2]_\mathcal{G} = P^\mathcal{G}_{\phi_2}$ as induction hypotheses:

1. Suppose $\phi = \neg\phi_1$. By Def. 10, for every state s, $s \in [\neg\phi_1]_\mathcal{G}$ iff $s \notin [\phi_1]_\mathcal{G}$. By induction hypothesis, $s \notin [\phi_1]_\mathcal{G}$ iff $s \notin P^\mathcal{G}_{\phi_1}$, and by Def. 17, $s \notin P^\mathcal{G}_{\phi_1}$ iff $s \in P^\mathcal{G}_{\neg\phi_1}$. Therefore, $[\neg\phi_1]_\mathcal{G} = P^\mathcal{G}_{\neg\phi_1}$.
2. Suppose $\phi = \phi_1 \wedge \phi_2$. By Def. 10, for every state s, $s \in [\phi_1 \wedge \phi_2]_\mathcal{G}$ iff $s \in [\phi_1]_\mathcal{G}$ and $s \in [\phi_2]_\mathcal{G}$. By induction hypothesis, $s \in [\phi_1]_\mathcal{G}$ and $s \in [\phi_2]_\mathcal{G}$ iff $s \in P^\mathcal{G}_{\phi_1}$ and $s \in P^\mathcal{G}_{\phi_2}$, and by Def. 17, $s \in P^\mathcal{G}_{\phi_1}$ and $s \in P^\mathcal{G}_{\phi_2}$ iff $s \in P^\mathcal{G}_{\phi_1 \wedge \phi_2}$. Therefore, $[\phi_1 \wedge \phi_2]_\mathcal{G} = P^\mathcal{G}_{\phi_1 \wedge \phi_2}$.

□

A result similar to Lemma 4 can be proved for general ATL-live formulas. A key difference, however, is that now the set $[\varphi]_\mathcal{G}$ is a subset of $P^\mathcal{G}_\varphi$ rather than being equal. This is because the constraints that are added to $\mathcal{D}$ by *theory* do not completely characterize $[\varphi]_\mathcal{G}$: they are necessary but not sufficient. As a result, the set $P^\mathcal{G}_\varphi$ includes $[\varphi]_\mathcal{G}$ and possibly some other states.

Lemma 5 *Let $\mathcal{D}$ be a declarative model and φ an ATL-live formula. Then,*

$$\textit{for all } \mathcal{G} \in CK(\mathit{theory}(\mathcal{D}, \varphi)),\ [\varphi]_\mathcal{G} \subseteq P^\mathcal{G}_\varphi$$

Proof. The proof is by induction on the structure of φ. In the following cases, we assume that $\mathcal{G} \in CK(\mathit{theory}(\mathcal{D}, \varphi))$.
Base case: suppose that φ is a propositional formula ϕ. By Lemma 4 we have $[\varphi]_\mathcal{G} = P^\mathcal{G}_\varphi$. In particular, $[\varphi]_\mathcal{G} \subseteq P^\mathcal{G}_\varphi$.
Induction step: according to the structure of φ, six cases are distinguished, with $[\varphi_1]_\mathcal{G} \subseteq P^\mathcal{G}_{\varphi_1}$ and $[\varphi_2]_\mathcal{G} \subseteq P^\mathcal{G}_{\varphi_2}$ as induction hypotheses:

1. Suppose $\varphi = \varphi_1 \vee \varphi_2$. By Def. 10, for every s, $s \in [\varphi_1 \vee \varphi_2]_{\mathcal{G}}$ iff $s \in [\varphi_1]_{\mathcal{G}}$ $s \in [\varphi_2]_{\mathcal{G}}$. By the induction hypotheses, if $s \in [\varphi_1]_{\mathcal{G}}$ (resp. $s \in [\varphi_2]_{\mathcal{G}}$) then $s \in P^{\mathcal{G}}_{\varphi_1}$ (resp. $s \in P^{\mathcal{G}}_{\varphi_2}$), and by Def. 17, $s \in P^{\mathcal{G}}_{\varphi_1}$ or $s \in P^{\mathcal{G}}_{\varphi_2}$ iff $s \in P^{\mathcal{G}}_{\varphi_1 \vee \varphi_2}$. Therefore, $[\varphi_1 \vee \varphi_2]_{\mathcal{G}} \subseteq P^{\mathcal{G}}_{\varphi_1 \vee \varphi_2}$.
2. The case for $\varphi = \varphi_1 \wedge \varphi_2$ is similar to (1).
3. Suppose $\varphi = \langle\langle\Gamma\rangle\rangle X\varphi'$. By the semantics of ATL, for every s, $s \in [\langle\langle\Gamma\rangle\rangle X\varphi']_{\mathcal{G}}$ iff for some actions $\alpha_1, \ldots, \alpha_{|\Gamma|}$, for all actions $\alpha_{|\Gamma|+1}, \ldots, \alpha_{|Ag\setminus\Gamma|}$, we have $\tau(s, \alpha_1, \ldots, \alpha_{|Ag|}) = s'$ and $s' \in [\varphi']_{\mathcal{G}}$. By induction hypothesis, if $\tau(s, \alpha_1, \ldots, \alpha_{|Ag|}) = s'$ and $s' \in [\varphi']_{\mathcal{G}}$, then $T(s, \alpha_1, \ldots, \alpha_{|Ag|}, s')$ and $s' \in P^{\mathcal{G}}_{\varphi'}$. By Def. 17, $\exists \alpha_1, \ldots, \alpha_{|\Gamma|} \forall \alpha_{|\Gamma|+1}, \ldots, \alpha_{|Ag\setminus\Gamma|}, T(s, \alpha_1, \ldots, \alpha_{|Ag|}, s')$ and $s' \in P^{\mathcal{G}}_{\varphi'}$ implies $s \in P^{\mathcal{G}}_{\langle\langle\Gamma\rangle\rangle X\varphi'}$. Therefore, $[\langle\langle\Gamma\rangle\rangle X\varphi']_{\mathcal{G}} \subseteq P^{\mathcal{G}}_{\langle\langle\Gamma\rangle\rangle X\varphi'}$.
4. Suppose $\varphi = \langle\langle\Gamma\rangle\rangle \varphi_1 U \varphi_2$. By the semantics of ATL, $s \in [\langle\langle\Gamma\rangle\rangle \varphi_1 U \varphi_2]_{\mathcal{G}}$ iff for some strategy F_Γ, for all outcomes $p \in out(s, F_\Gamma)$, there exists $j \geq 1$, such that $s_j \in [\varphi_2]_{\mathcal{G}}$, and for all $i < j$, $s_i \in [\varphi_1]_{\mathcal{G}}$. We prove by induction on j, the least number of steps required to get to a state that satisfies φ_2, to prove that $s \in P^{\mathcal{G}}_{\langle\langle\Gamma\rangle\rangle \varphi_1 U \varphi_2}$. We assume that $s \in [\langle\langle\Gamma\rangle\rangle \varphi_1 U \varphi_2]_{\mathcal{G}}$:
   Base case: assume $j = 0$. In this case, $s \in [\varphi_2]_{\mathcal{G}}$. By the induction hypothesis, we have $s \in P^{\mathcal{G}}_{\varphi_2}$, and according to (i) in Def. 17, line 7, we obtain $s \in P^{\mathcal{G}}_{\langle\langle\Gamma\rangle\rangle \varphi_1 U \varphi_2}$, as required.
   Induction step: assume $j = m + 1$. The induction hypothesis for this inner induction is: if $s' \in [\langle\langle\Gamma\rangle\rangle \varphi_1 U \varphi_2]_{\mathcal{G}}$ and from s' we can reach a state satisfying φ_2 in less than m steps, then $s' \in P^{\mathcal{G}}_{\langle\langle\Gamma\rangle\rangle \varphi_1 U \varphi_2}$. Since j is the least number of steps required to get to a state that satisfies φ_2, state s does not satisfy φ_2 itself. Hence, there exists a state s' next to s that satisfies $\langle\langle\Gamma\rangle\rangle \varphi_1 U \varphi_2$ and j for that state is less than m: $\exists s' T(s, \alpha_1, \ldots, \alpha_{|Ag|}, s')$ and $s' \in [\langle\langle\Gamma\rangle\rangle \varphi_1 U \varphi_2]_{\mathcal{G}}$, where for all $a \in \Gamma$, $\alpha_a = F_a(s)$. According to the induction hypothesis for the inner induction, we have $\exists s'(T(s, \alpha_1, \ldots, \alpha_{|Ag|}, s')$ and $s' \in P^{\mathcal{G}}_{\langle\langle\Gamma\rangle\rangle \varphi_1 U \varphi_2}$. Since $s \in [\langle\langle\Gamma\rangle\rangle \varphi_1 U \varphi_2]_{\mathcal{G}}$ and $s \notin [\varphi_2]_{\mathcal{G}}$, by the semantics of ATL, we conclude that $s \in [\varphi_1]_{\mathcal{G}}$. Using the induction hypothesis for the outer induction ($[\varphi_1]_{\mathcal{G}} \subseteq P^{\mathcal{G}}_{\varphi_1}$), we derive $s \in P^{\mathcal{G}}_{\varphi_1}$. Finally, according to (ii) in Def. 17, line 7, and the above property, we obtain $s \in P^{\mathcal{G}}_{\langle\langle\Gamma\rangle\rangle \varphi_1 U \varphi_2}$. Thus, if $j = m + 1$ then $s \in P^{\mathcal{G}}_{\langle\langle\Gamma\rangle\rangle \varphi_1 U \varphi_2}$. By putting the base case and the induction step together, we conclude that $[\langle\langle\Gamma\rangle\rangle \varphi_1 U \varphi_2]_{\mathcal{G}} \subseteq P^{\mathcal{G}}_{\langle\langle\Gamma\rangle\rangle \varphi_1 U \varphi_2}$.
5. The case for $\varphi = [\![\Gamma]\!] X\varphi'$ is similar to (3).
6. The case for $\varphi = [\![\Gamma]\!] \varphi_1 U \varphi_2$ is similar to (4).

□

As a result, Lemma 5 extends Lemma 4 to the whole ATL-live, but we only have inclusion and not equality between the interpretations of formula φ and relation symbol P_φ.

The next lemma relates every CGS in $CK(\mathcal{D})$ to a CGS in $CK(\textit{theory}(\mathcal{D}, \varphi))$.

Lemma 6 *Let $\mathcal{D}$ be a declarative model and φ an ATL-live formula. For every $\mathcal{G} \in CK(\mathcal{D})$ there exists $\mathcal{G}' \in CK(\textit{theory}(\mathcal{D}, \varphi))$ such that $s'_0 = s_0$ and $P^{\mathcal{G}'}_\varphi = [\varphi]_{\mathcal{G}}$.*

Proof. Suppose $\mathcal{G} \in CK(\mathcal{D})$. Let $\mathcal{G}'$ be an interpretation with the same domain as $\mathcal{G}$. For each symbol in the base of $\mathcal{D}$, $\mathcal{G}'$ has the same value as $\mathcal{G}$, and for every subformula φ' of φ, $\mathcal{G}'$ assigns $[\varphi']_{\mathcal{G}}$ to symbol $P_{\varphi'}$, thus $P^{\mathcal{G}'}_{\varphi'} = [\varphi']_{\mathcal{G}}$. According to the semantics of ATL, the constraints that are added to $\mathcal{D}$ for each subformula φ' of φ by function *theory*, are satisfied by sets $[\varphi']_{\mathcal{G}}$. Thus, $\mathcal{G}'$ is a model of $CK(\textit{theory}(\mathcal{D}, \varphi))$, i.e., $\mathcal{G}' \in CK(\textit{theory}(\mathcal{D}, \varphi))$. □

In the next theorem we present our main contribution by combining the results we have proved so far: universal model checking of ATL-live formulas can be reduced to checking semantic entailment in FOL.

Theorem 7 *Let $\mathcal{D}$ be a declarative model and φ a ATL-live formula. Then,*

$$\mathcal{D} \models_\forall \varphi \textit{ iff } \textit{theory}(\mathcal{D}, \varphi) \models \forall s(S_0(s) \rightarrow P_\varphi(s))$$

Proof. We prove that (i) if $\mathcal{D} \models_\forall \varphi$ then $\textit{theory}(\mathcal{D}, \varphi) \models \forall s(S_0(s) \rightarrow P_\varphi(s))$, and (ii) if $\mathcal{D} \not\models_\forall \varphi$ then $\textit{theory}(\mathcal{D}, \varphi) \not\models \forall s(S_0(s) \rightarrow P_\varphi(s))$.

(i) Assume $\mathcal{D} \models_\forall \varphi$. We show that every model of $\textit{theory}(\mathcal{D}, \varphi)$ also satisfies $\forall s(S_0(s) \rightarrow P_\varphi(s))$, that is, for all $\mathcal{G} \in CK(\textit{theory}(\mathcal{D}, \varphi))$, $s_0 \in P^{\mathcal{G}}_\varphi$. By Lemma 3, for every $\mathcal{G} \in CK(\textit{theory}(\mathcal{D}, \varphi))$, there exists $\mathcal{G}' \in CK(\mathcal{D})$ that it is a substructure of $\mathcal{G}$. Since $\mathcal{G}' \in CK(\mathcal{D})$ and $\mathcal{D} \models_\forall \varphi$, we have that $\mathcal{G}' \models \varphi$. Since $\mathcal{G}'$ is a substructure of $\mathcal{G}$, $\mathcal{G}' \models \varphi$ implies $\mathcal{G} \models \varphi$, and by the semantics of ATL, $\mathcal{G} \models \varphi$ implies $s_0 \in [\varphi]_{\mathcal{G}}$. Then, by Lemma 5, $s_0 \in [\varphi]_{\mathcal{G}}$ implies $s_0 \in P^{\mathcal{G}}_\varphi$. As a result, for every $\mathcal{G}' \in CK(\textit{theory}(\mathcal{D}, \varphi))$, $s_0 \in P^{\mathcal{G}}_\varphi$ as required.

(ii) Assume $\mathcal{D} \not\models_\forall \varphi$. We show that there exists a model of $\textit{theory}(\mathcal{D}, \varphi)$ that does not satisfy $\forall s(S_0(s) \rightarrow P_\varphi(s))$, that is, there exists $\mathcal{G} \in CK(\textit{theory}(\mathcal{D}, \varphi))$ such that $s_0 \notin P^{\mathcal{G}}_\varphi$. Since $\mathcal{D} \not\models_\forall \varphi$, there exists a CGS $\mathcal{G} \in CK(\mathcal{D})$ that does not satisfy φ, that is, $s_0 \notin [\varphi]_{\mathcal{G}}$. Since $\mathcal{G} \in CK(\mathcal{D})$, by Lemma 6, there exists a CGS $\mathcal{G}' \in CK(\textit{theory}(\mathcal{D}, \varphi))$ such that $s_0 = s'_0$ and $[\varphi]_{\mathcal{G}} = P^{\mathcal{G}'}_\varphi$. Since $s_0 \notin P^{\mathcal{G}}_\varphi$, we have that $s'_0 \notin P^{\mathcal{G}'}_\varphi$. Thus, we obtain that for $\mathcal{G}' \in CK(\textit{theory}(\mathcal{D}, \varphi))$, $s'_0 \notin P^{\mathcal{G}'}_\varphi$ as required. □

By Theorem 7 we can reduce a universal model checking instance $\mathcal{D} \models_\forall \varphi$ to verifying the entailment between $\textit{theory}(\mathcal{D}, \varphi)$ and $\forall s(S_0(s) \rightarrow P_\varphi(s))$. In principle, the latter check can be performed automatically by an SMT solver.

We conclude this section by elaborating on the existential model checking problem in Def. 15. To solve this latter problem using FOL reasoning, we remark that a CGS $\mathcal{G}$ satisfies an ATL formula $\neg\varphi$ iff $\mathcal{G}$ does not satisfy φ. Note that the coimplication holds because we consider CGS with a single initial state. More formally, we state next result, which follows immediately by the semantics of ATL.

Lemma 8 *Let $\mathcal{G}$ be a CGS and φ an ATL formula. Then,*

$$\mathcal{G} \models \neg\varphi \textit{ iff } \mathcal{G} \not\models \varphi$$

By Lemma 8, we can prove the following corollary:

Corollary 9 *Let $\mathcal{D}$ be a declarative model and φ an ATL-live formula. Then,*

$$\mathcal{D} \models_\exists \neg\varphi \textit{ iff } \mathcal{D} \not\models_\forall \varphi \textit{ iff } \textit{theory}(\mathcal{D}, \varphi) \not\models P_\varphi(s_0)$$

By Corollary 9 the existential model checking of a negated ATL-live formula can be reduced to universal model checking.

5 Maximality of ATL-live

In Theorem 7 we showed that model checking ATL-live can be reduced to semantic entailment in FOL. However, the logical connectives $\langle\langle A\rangle\rangle G$, $[\![A]\!] G$, and $\neg$ over temporal connectives are not included in ATL-live. We show that model checking these three connectives is not reducible to FOL entailment, by reducing the complement of the halting problem on an empty tape for a deterministic

Turing machines to universal model checking of formulas of type $\langle\langle Ag\rangle\rangle G\varphi$ and $[\![Ag]\!]G\varphi$. The complement of the halting problem is not recursively enumerable, but FOL entailment is. Therefore, universal model checking of $\langle\langle Ag\rangle\rangle G$ and $[\![Ag]\!]G$ cannot be reduced to checking entailment in FOL. We call this result the *maximality* of ATL-live. We start by introducing deterministic Turing machines.

Definition 18 (DTM) *A* deterministic Turing machines *is a tuple* $M = \langle \mathcal{V}, \Sigma, \delta\rangle$*, where* $\mathcal{V} = \{v_1, \ldots, v_n\}$ *is a finite set of states,* $\Sigma = \{B, 0\}$ *is the tape alphabet, and* δ *is the transition function from* $\mathcal{V} \times \Sigma$ *to* $\mathcal{V} \times \Sigma \times \{L, R\}$.

A DTM $M = \langle \mathcal{V}, \Sigma, \delta\rangle$ is assumed to start in state v_0. We consider M to have halted if it reaches state v_n. The tape is one way infinite. In the initial state, the read/write head is on the left-most square of the tape, and every square on the tape is blank (B).

The intuition behind reducing the complement of the halting problem on an empty tape for a DTM to universal model checking of $\langle\langle Ag\rangle\rangle G$ and $[\![Ag]\!]G$ is that the set of all the configurations of a DTM can be seen as the state space for a CGS and the transition relation of this CGS can be derived from the transition function of the DTM. Since the underlying DTM is deterministic, this CGS has only one computation path, and therefore, satisfying $\langle\langle Ag\rangle\rangle G$ and $[\![Ag]\!]G$ is equivalent.

Lemma 10 *Let* $M = \langle \mathcal{V}, \Sigma, \delta\rangle$ *be a* DTM*. The complement of the halting problem on an empty tape for* M *is reducible to universal model checking of a formula* $\langle\langle Ag\rangle\rangle G$.

Proof. To prove this result, we define a declarative model $\mathcal{D}_M$ based on M such that $\mathcal{D}_M$ universally satisfies formula $\langle\langle Ag\rangle\rangle G\neg halt$ iff M does not halt on an empty tape. To encode M as a declarative model $\mathcal{D}_M = \langle \mathcal{B}, \Gamma\rangle$, we use the base $\mathcal{B} = \langle \mathcal{F}, \mathcal{R}\rangle$ such that

- $\mathcal{F} = \{0, inc/1, dec/1, V/1, H/1\}$[5];
- $\mathcal{R} = \{B/2, S_0/1, T/|Ag| + 2, halt/1\}$.

The constant 0 represents the corresponding number. The function symbols $inc/1$ and $dec/1$ are used to model increment and decrement on natural numbers. We can refer to a natural number n by applying n times inc to 0. In this lemma and the following, natural numbers are shorthands of their representations using base $\mathcal{B}$. Natural numbers are used to represent configurations of M: the position of the head, the current state of M, and to point to different squares on the tape. The expression $V(t) = i$ intuitively represents that the state of M at step t is v_i, and $H(t) = i$ represents that the head of M at step t is on the i-th square. The binary relation symbol $B(t, i)$ holds if at step t the i-th square is blank. The relation symbols S_0 and T are used to model the initial state and transition function, while $halt$ is a relation symbol to represent the halting state.

In the declarative model $\mathcal{D}_M = \langle \mathcal{B}, \Gamma\rangle$, the constraints in Γ consist of 5 parts:

1. Formulas to encode the intended semantics of 0, inc, and dec:
   - $\forall i(inc(i) \neq 0)$
   - $\forall i, i'(inc(i) = inc(i') \rightarrow i = i')$
   - $\forall i(i \neq 0 \rightarrow (\exists i'(inc(i') = i)))$
   - $dec(0) = 0$
   - $\forall i(dec(inc(i)) = i)$
   - $\forall i(i \neq 0 \rightarrow inc(dec(i)) = i)$.
2. A formula stating that at each step of computation at most one position of the tape can be changed: $\forall t, i(H(t) \neq i \rightarrow (B(t, i) \leftrightarrow B(inc(t), i)))$.
3. Formulas to encode the initial configuration of M.
   - $V(0) = 0$: at step 0, M is at state v_0;
   - $H(0) = 0$: at step 0, the tape head of M is at position 0;
   - $\forall i B(0, i)$: at step 0, every position of the tape is blank.
4. Formulas to encode the transition function δ: for every pair in $\mathcal{V} \times \Sigma$ we have a formula that mimics the computation of M.
5. Formulas for the initial state, transition function, and halting state of the corresponding CGS. We use natural numbers as the state space of the CGS. The configuration of M at state (step) t is represented by $V(t)$, $H(t)$, and $B(t, \cdot)$:
   - The initial state: $\forall t(S_0(t) \leftrightarrow t = 0)$;
   - The transition function: $\forall t, t', x_1, \ldots, x_{|Ag|}(T(t, x_1, \ldots, x_{|Ag|}, t') \leftrightarrow t' = inc(t))$;
   - The halting states: $\forall t(halt(t) \leftrightarrow V(t) = n)$.

We now claim that the following holds:

$\mathcal{D}_M \models_\forall \langle\langle Ag\rangle\rangle G\neg halt$ iff M does not halt on an empty tape.

($\Rightarrow$) $\mathcal{D}_M \models_\forall \langle\langle Ag\rangle\rangle G\neg halt$ implies that every CGS $\mathcal{G} \in CK(\mathcal{D}_M)$ satisfies $\langle\langle Ag\rangle\rangle G\neg halt$. The standard interpretation of natural numbers, which satisfies $\mathcal{D}_M$, corresponds to the computation of M. Since $\langle\langle Ag\rangle\rangle G\neg halt$ means that there exists a path along which halt is never true, and the DTM M is deterministic with only one path, we can conclude that M does not halt on an empty tape.

($\Leftarrow$) By induction on the number of steps, we can prove that if at step t, M is at state v_i, every CGS $\mathcal{G} \in CK(\mathcal{D}_M)$ satisfies $V(t) = i$. Assuming M does not halt on an empty tape, we can conclude that every CGS $\mathcal{G}$ in $CK(\mathcal{D}_M)$ has an infinite path $0, 1, 2, \ldots$, where none of them is the halting state v_n. Therefore, every $\mathcal{G} \in CK(\mathcal{D}_M)$ satisfies $\langle\langle Ag\rangle\rangle G\neg halt$, that is, $\mathcal{D}_M \models_\forall \langle\langle Ag\rangle\rangle G\neg halt$. □

Next result can be proved by using the same construction and reduction as for Lemma 10.

Lemma 11 *Let* $M = \langle \mathcal{V}, \Sigma, \delta\rangle$ *be a DTM. The complement of the halting problem on an empty tape for* M *is reducible to universal model checking of a formula* $[\![Ag]\!]G$.

We conclude this section by stating our maximality result.

Theorem 12 (Maximality of ATL-live) *The temporal part of ATL-live cannot be extended with $\langle\langle A\rangle\rangle G$, $[\![A]\!]G$, or $\neg$ over temporal objectives, for universal model checking in FOL.*

6 Conclusions

In this paper we presented ATL-live, a fragment of ATL for which the model checking problem is reducible to semantic entailment in FOL. ATL-live comprises two parts: propositional and temporal. The propositional part contains all Boolean connectives, whereas the temporal part includes all strategic operators that are normally used to express liveness properties. Our model checking technique accepts as input a set of formulas, called a *declarative model*, where every satisfying interpretation is a CGS. As a result, a declarative model

[5] For simplicity, we make use of function symbols in the reduction. However, notice that these are just shorthands, as they can be expressed by using relation symbols and identity, thus conforming to Def. 13.

represents in general a class of CGSs. In this setting, we studied two decision problems: universal and existential model checking. In universal (resp. existential) model checking we want to check whether all (resp. some) CGSs in the relevant class satisfy a given ATL formula. We showed how our encoding of ATL-live in FOL can be used to solve universal model checking and how existential model checking can be reduced to the latter in some cases. Finally, we proved that ATL-live is maximal in the sense that if any other ATL connective is added, non-FOL reasoning techniques would be required.

As future work, we plan to study the use of SMT solvers and decidable fragments of FOL for model checking ATL-live formulas. After this step, we envisage to extend our framework in two different and interesting directions: to infinite models (such as data-aware systems [5]) and formulas with arithmetic operators.

ACKNOWLEDGEMENTS

F. Belardinelli acknowledges the support of ANR JCJC Project SVeDaS (ANR-16-CE40-0021).

REFERENCES

[1] R. Alur, T. Henzinger, F. Mang, S. Qadeer, S. Rajamani, and S. Tasiran, 'MOCHA: Modularity in model checking', in *Proceedings of the 10th International Conference on Computer Aided Verification (CAV98)*, volume 1427 of *LNCS*, pp. 521–525. Springer-Verlag, (1998).

[2] R. Alur, T. A. Henzinger, and O. Kupferman, 'Alternating-time temporal logic', *Journal of the ACM*, **49**(5), 672–713, (2002).

[3] C. Areces, P. Fontaine, and S. Merz, 'Modal satisfiability via smt solving', in *Software, Services, and Systems*, 30–45, Springer, (2015).

[4] F. Belardinelli, R. Condurache, C. Dima, W. Jamroga, and A. V. Jones, 'Bisimulations for verifying strategic abilities with an application to threeballot', in *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2017, São Paulo, Brazil, May 8-12, 2017*, eds., Kate Larson, Michael Winikoff, Sanmay Das, and Edmund H. Durfee, pp. 1286–1295. ACM, (2017).

[5] F. Belardinelli and V. Malvone, 'Decidable verification of agent-based data-aware systems', in *PRIMA 2019: Principles and Practice of Multi-Agent Systems - 22nd International Conference, Turin, Italy, October 28-31, 2019, Proceedings*, pp. 52–68, (2019).

[6] N. Bulling, J. Dix, and W. Jamroga, 'Model checking logics of strategic ability: Complexity', in *Specification and Verification of Multi-agent Systems*, 125–159, Springer, (2010).

[7] D. Calvanese, S. Ghilardi, A. Gianola, M. Montali, and A. Rivkin, 'Formal modeling and SMT-based parameterized verification of multi-case data-aware bpmn', *arXiv preprint arXiv:1905.12991*, (2019).

[8] A. Church, 'An unsolvable problem of elementary number theory', *American journal of mathematics*, **58**(2), 345–363, (1936).

[9] E. M. Clarke and E. Emerson, 'Design and synthesis of synchronization skeletons for branching-time temporal logic', in *Proceedings of Workshop on Logic of Programs*, volume 131 of *LNCS*, pp. 52–71. Springer-Verlag, (1981).

[10] L. De Moura and N. Bjørner, 'Satisfiability modulo theories: introduction and applications', *Communications of the ACM*, **54**(9), 69–77, (2011).

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

[12] J. Ezekiel, A. Lomuscio, L. Molnar, and S. Veres, 'Verifying fault tolerance and self-diagnosability of an autonomous underwater vehicle', in *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI11)*, pp. 1659–1664. AAAI Press, (2011).

[13] R. Fagin, J. Y. Halpern, Y. Moses, and M. Y. Vardi, *Reasoning about Knowledge*, MIT Press, Cambridge, 1995.

[14] P. Gammie and R. van der Meyden, 'MCK: Model checking the logic of knowledge', in *Proceedings of 16th International Conference on Computer Aided Verification (CAV04)*, volume 3114 of *Lecture Notes in Computer Science*, pp. 479–483. Springer, (2004).

[15] S. Ghilardi and S. Ranise, 'Mcmt: A model checker modulo theories', in *International Joint Conference on Automated Reasoning*, pp. 22–29. Springer, (2010).

[16] N. Gorogiannis, F. Raimondi, and I. Boureanu, 'A novel symbolic approach to verifying epistemic properties of programs'. International Joint Conferences on Artificial Intelligence, (2017).

[17] E. Grädel, P. Kolaitis, and M. Vardi, 'On the decision problem for two-variable first-order logic', *Bulletin of Symbolic Logic*, **3**(1), 53–69, (1997).

[18] E. Grädel, 'On the restraining power of guards', *Journal of Symbolic Logic*, **64**(4), 1719–1742, (1999).

[19] M. R. A. Huth and M. D. Ryan, *Logic in Computer Science: Modelling and Reasoning about Systems (2nd edition)*, Cambridge University Press, Cambridge, England, 2004.

[20] W. Jamroga and W. van der Hoek, 'Agents that know how to play', *Fundamenta Informaticae*, **62**, 1–35, (2004).

[21] W. Jamroga, M. Knapik, and D. Kurpiewski, 'Model checking the SELENE e-voting protocol in multi-agent logics', in *Electronic Voting - Third International Joint Conference, E-Vote-ID 2018, Bregenz, Austria, October 2-5, 2018, Proceedings*, eds., Robert Krimmer, Melanie Volkamer, Véronique Cortier, Rajeev Goré, Manik Hapsara, Uwe Serdült, and David Duenas-Cid, volume 11143 of *Lecture Notes in Computer Science*, pp. 100–116. Springer, (2018).

[22] M. Kacprzak, W. Nabialek, A. Niewiadomski, W. Penczek, A. Pólrola, M. Szreter, B. Woźna, and A. Zbrzezny, 'Verics 2007 - a model checker for knowledge and real-time', *Fundamenta Informaticae*, **85**(1), 313–328, (2008).

[23] C. Baier J. P. Katoen, *Principles of Model Checking (Representation and Mind Series)*, The MIT Press, 2008.

[24] P. Kouvaros and A. Lomuscio, 'Parameterised verification for multi-agent systems', *Artificial Intelligence*, **234**, 152–189, (2016).

[25] J. Li, S. Zhu, G. Pu, and M. Y. Vardi, 'Sat-based explicit ltl reasoning', in *Haifa Verification Conference*, pp. 209–224. Springer, (2015).

[26] A. Lomuscio, H. Qu, and F. Raimondi, 'MCMAS: A model checker for the verification of multi-agent systems', *Software Tools for Technology Transfer*, (2015). http://dx.doi.org/10.1007/s10009-015-0378-x.

[27] A. Lomuscio and J. Michliszyn, 'Verification of multi-agent systems via predicate abstraction against ATLK specifications', in *Proceedings of the 2016 International Conference on Autonomous Agents & Multi-agent Systems*, pp. 662–670. International Foundation for Autonomous Agents and Multiagent Systems, (2016).

[28] J.-J. Ch. Meyer and W. van der Hoek, *Epistemic Logic for AI and Computer Science*, volume 41 of *Cambridge Tracts in Theoretical Computer Science*, Cambridge University Press, 1995.

[29] F. Mogavero, A. Murano, G. Perelli, and M. Y. Vardi, 'Reasoning about strategies: On the model-checking problem', *ACM Transactions in Computational Logic*, **15**(4), 34:1–34:47, (2014).

[30] A. Pnueli, 'The temporal logic of programs', in *Proceedings of the 18th International Symposium Foundations of Computer Science (FOCS77)*, pp. 46–57, (1977).

[31] Ph. Schnoebelen, 'The complexity of temporal logic model checking', in *Proceedings of the 4th Conference Advances in Modal Logic (AiML02)*, volume 4 of *Advances in Modal Logic*, 437–459, King's College Publications, (2003).

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

[33] S. Tonetta, 'Linear-time temporal logic with event freezing functions', *arXiv preprint arXiv:1709.02103*, (2017).

[34] A. Vakili and N. A. Day, 'Temporal logic model checking in alloy', in *International Conference on Abstract State Machines, Alloy, B, VDM, and Z*, pp. 150–163. Springer, (2012).

[35] A. Vakili and N. A. Day, 'Reducing CTL-live model checking to first-order logic validity checking', in *Proceedings of the 14th Conference on Formal Methods in Computer-Aided Design*, pp. 215–218. FMCAD Inc, (2014).

[36] A. Vakili and N. A. Day, 'Reducing CTL-live model checking to semantic entailment in first-order logic (version 1)', *Cheriton School of Comp. Sci., University of Waterloo, Tech. Rep. CS-2014-05*, (2014).

[37] A. Vakili and N. A. Day, 'Verifying CTL-live properties of infinite state models using an smt solver', in *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*, pp. 213–223. ACM, (2014).

[38] M. Wooldridge, *An introduction to MultiAgent systems*, Wiley, second edition edn., 2009.