On the Expressiveness and Complexity of ATL

François Laroussinie, Nicolas Markey, and Ghassan Oreiby

LSV, CNRS & ENS Cachan, France

Abstract. ATL is a temporal logic geared towards the specification and verification of properties in multi-agents systems. It allows to reason on the existence of strategies for coalitions of agents in order to enforce a given property. We prove that the standard definition of ATL (built on modalities "Next", "Always" and "Until") has to be completed in order to express the duals of its modalities: it is necessary to add the modality "Release". We then precisely characterize the complexity of ATL model-checking when the number of agents is not fixed. We prove that it is Δ_2^{P}- and Δ_3^{P}-complete, depending on the underlying multi-agent model (ATS and CGS resp.). We also prove that ATL^+ model-checking is Δ_3^{P}-complete over both models, even with a fixed number of agents.

1 Introduction

Model checking. Temporal logics were proposed for the specification of reactive systems almost thirty years ago [16]. They have been widely studied and successfully used in many situations, especially for model checking —the automatic verification that a finite-state model of a system satisfies a temporal logic specification. Two flavors of temporal logics have mainly been studied: *linear-time temporal logics*, *e.g.* LTL [16], which expresses properties on the possible *executions* of the model; and *branching-time temporal logics*, such as CTL [7, 17], which can express requirements on *states* (which may have several possible futures) of the model.

Alternating-time temporal logic. Over the last ten years, a new flavor of temporal logics has been defined: *alternating-time temporal logics*, *e.g.* ATL [2, 3]. ATL is a fundamental logic for verifying properties in *synchronous multi-agent systems*, in which several agents can concurrently influence the behavior of the system. This is particularly interesting for modeling control problems. In that setting, it is not only interesting to know if something *can arrive* or *will arrive*, as can be expressed in CTL or LTL, but rather if some agent(s) can *control* the evolution of the system in order to enforce a given property.

The logic ATL can precisely express this kind of properties, and can for instance state that "there is a strategy for a coalition A of agents in order to eventually reach an accepting state, whatever the other agents do". ATL is an extension of CTL, its formulae are built on atomic propositions and boolean combinators, and (following the seminal papers [2, 3]) on modalities $\langle\!\langle A\rangle\!\rangle\, \mathbf{X}\, \varphi$ (coalition A has a strategy to immediately enter a state satisfying φ), $\langle\!\langle A\rangle\!\rangle\, \mathbf{G}\, \varphi$ (coalition A can force the system to always satisfy φ) and $\langle\!\langle A\rangle\!\rangle\, \varphi\, \mathbf{U}\, \psi$ (coalition A has a strategy to enforce $\varphi\, \mathbf{U}\, \psi$).

Multi-agent models. While linear- and branching-time temporal logics are interpreted on Kripke structure, alternating-time temporal logics are interpreted on models that incorporate the notion of *multiple agents*. Two kinds of synchronous multi-agent models have been proposed for ATL in the literature. First *Alternating Transition Systems* (ATSs)[2] have been defined: in any location of an ATS, each agent chooses one *move*, *i.e.*, a subset of locations (the list of possible moves is defined explicitly in the model) in which he would like the execution to go to. When all the agents have made their choice, the intersection of their choices is required to contain one single location, in which the execution enters. In the second family of models, called *Concurrent Game Structures* (CGSs) [3], each of the n agents has a finite number of possible moves (numbered with integers), and, in each location, an n-ary transition function indicates the state to which the execution goes.

Our contributions. While in LTL and CTL, the dual of "Until" modality can be expressed as a disjunction of "always" and "until", we prove that it is not the case in ATL. In other words, ATL, as defined in [2, 3], is not as expressive as one could expect (while it is known that adding the dual of "Until" does not increase the complexity of the verification problems [5, 10]).

We also precisely characterize the complexity of the model checking problem. The original works about ATL provide model-checking algorithms in time $O(m \cdot l)$, where m is the number of transitions in the model, and l is the size of the formula [2, 3], thus in PTIME. However, contrary to Kripke structures, the number of transitions in a CGS or in an ATS is not quadratic in the number of states [3], and might even be exponential in the number of agents. PTIME-completeness thus only holds for ATS when the number of agents is bounded, and it is shown in [12, 13] that the problem is strictly[1] harder otherwise, namely NP-hard on ATS and Σ_2^{P}-hard on CGSs where the transition function is encoded as a boolean function. We prove that it is in fact Δ_2^{P}-complete and Δ_3^{P}-complete, resp., correcting wrong algorithms in [12, 13] (the problem lies in the way the algorithms handle negations). We also show that ATL^+ is Δ_3^{P}-complete on both ATSs and CGSs, even when the number of agents is fixed, extending a result of [18]. Finally we study translations between ATS and CGS.

Related works. In [2, 3] ATL has been proposed and defined over ATS and CGS. In [11] expressiveness issues are considered for ATL^* and ATL. Complexity of satisfiability is addressed in [10, 19]. Complexity results about model checking (for ATL, ATL^+, ATL^*) can be found in [3, 18]. Regarding control- and game theory, many papers have focused on this wide area; we refer to [20] for a survey, and to its numerous references for a complete overview.

Plan of the paper. Section 2 contains the necessary formal definitions needed in the sequel. Section 3 explains our expressiveness result, and Section 4 deals with

[1] We adopt the classical hypothesis that the polynomial-time hierarchy does not collapse, and that PTIME $\neq$ NP. We refer to [15] for the definitions about complexity classes, especially about oracle Turing machines and the polynomial-time hierarchy.

the model-checking algorithms. Due to lack of space, some proofs are omitted in this article, but can be read in the technical appendix at the end of the paper.

2 Definitions

2.1 Concurrent Game Structures and Alternating Transition Systems

Definition 1. *A* Concurrent Game Structure *(*CGS *for short)* $\mathcal{C}$ *is a* 6*-tuple* $(\mathsf{Agt}, \mathsf{Loc}, \mathsf{AP}, \mathsf{Lab}, \mathsf{Mov}, \mathsf{Edg})$ *s.t:*

- $\mathsf{Agt} = \{A_1, ..., A_k\}$ *is a finite set of* agents *(or* players*);*
- Loc *and* AP *are two finite sets of* locations *and* atomic propositions*, resp.;*
- $\mathsf{Lab}\colon \mathsf{Loc} \to 2^{\mathsf{AP}}$ *is a function labeling each location by the set of atomic propositions that hold for that location;*
- $\mathsf{Mov}\colon \mathsf{Loc} \times \mathsf{Agt} \to \mathcal{P}(\mathbb{N}) \smallsetminus \{\varnothing\}$ *defines the (finite) set of possible moves of each agent in each location.*
- $\mathsf{Edg}\colon \mathsf{Loc} \times \mathbb{N}^k \to \mathsf{Loc}$*, where* $k = |\mathsf{Agt}|$*, is a (partial) function defining the transition table. With each location and each set of moves of the agents, it associates the resulting location.*

The intended behaviour is as follows [3]: in a given location ℓ, each player A_i chooses one possible move m_{A_i} in $\mathsf{Mov}(\ell, A_i)$ and the successor location is given by $\mathsf{Edg}(\ell, m_{A_1}, ..., m_{A_k})$. We write $\mathsf{Next}(\ell)$ for the set of all possible successor locations from ℓ, and $\mathsf{Next}(\ell, A_j, m)$ for the restriction of $\mathsf{Next}(\ell)$ to locations reachable from ℓ when player A_j makes the move m.

In the original works about ATL [2], the logic was interpreted on ATSs, which are transition systems slightly different from CGSs:

Definition 2. *An* Alternating Transition System *(*ATS *for short)* $\mathcal{A}$ *is a* 5*-tuple* $(\mathsf{Agt}, \mathsf{Loc}, \mathsf{AP}, \mathsf{Lab}, \mathsf{Mov})$ *where:*

- Agt*,* Loc*,* AP *and* Lab *have the same meaning as in CGSs;*
- $\mathsf{Mov}\colon \mathsf{Loc} \times \mathsf{Agt} \to \mathcal{P}(\mathcal{P}(\mathsf{Loc}))$ *associate with each location* ℓ *and each agent* a *the set of possible moves, each move being a subset of* Loc*. For each location* ℓ*, it is required that, for any* $Q_i \in \mathsf{Mov}(\ell, A_i)$*,* $\bigcap_{i \leq k} Q_i$ *be a singleton.*

The intuition is as follows: in a location ℓ, once all the agents have chosen their moves (*i.e.*, a subset of locations), the execution goes to the (only) state that belongs to all the sets chosen by the players. Again $\mathsf{Next}(\ell)$ (resp. $\mathsf{Next}(\ell, A_j, m)$) denotes the set of all possible successor locations (resp. the set of possible successor locations when player A_j chooses the move m).

We prove in Section 4.2 that both models have the same expressiveness (w.r.t. alternating bisimilarity [4]).

2.2 Strategy, outcomes of a strategy

Let $\mathcal{S}$ be a CGS or an ATS. A *computation* of $\mathcal{S}$ is an infinite sequence $\rho = \ell_0\ell_1\cdots$ of locations such that for any i, $\ell_{i+1} \in \mathsf{Next}(\ell_i)$. We can use the standard notions of suffix and prefix for these computations; $\rho[i]$ denotes the i-th location ℓ_i. A *strategy* for a player $A_i \in \mathsf{Agt}$ is a function f_{A_i} that maps any finite prefix of a computation to a possible move for A_i[2]. A strategy is *state-based* (or *memoryless*) if it only depends on the current state (*i.e.*, $f_{A_i}(\ell_0\cdots\ell_m) = f_{A_i}(\ell_m)$).

A strategy induces a set of computations from ℓ —called the *outcomes* of f_{A_i} from ℓ and denoted[3] $\mathsf{Out}_{\mathcal{S}}(\ell, f_{A_i})$— that player A_i can enforce: $\ell_0\ell_1\ell_2\cdots \in \mathsf{Out}_{\mathcal{S}}(\ell, f_{A_i})$ iff $\ell = \ell_0$ and for any i we have $\ell_{i+1} \in \mathsf{Next}(\ell_i, A_i, f_{A_i}(\ell_0\cdots\ell_i))$. Let $A \subseteq \mathsf{Agt}$ be a coalition. A strategy for A is a tuple F_A containing one strategy for every player in A: $F_A = \{f_{A_i} | A_i \in A\}$. The outcomes of F_A from a location ℓ contains the computations enforced by the strategies in F_A: $\ell_0\ell_1\cdots \in \mathsf{Out}_{\mathcal{S}}(\ell, F_A)$ s.t. $\ell = \ell_0$ and for any i, $\ell_{i+1} \in \bigcap_{A_i \in A} \mathsf{Next}(\ell_i, A_i, f_{A_i}(\ell_0\cdots\ell_i))$. The set of strategies for A is denoted[3] $\mathsf{Strat}_{\mathcal{S}}(A)$. Finally $\mathsf{Out}_{\mathcal{S}}(\ell, \emptyset)$ represents the set of all computations from ℓ.

2.3 The logic ATL and some extensions

Again, we follow the definitions of [3]:

Definition 3. *The syntax of ATL is defined by the following grammar:*

$$\mathsf{ATL} \ni \varphi_s, \psi_s ::= \top \mid p \mid \neg\varphi_s \mid \varphi_s \vee \psi_s \mid \langle\!\langle A \rangle\!\rangle\, \varphi_p$$
$$\varphi_p ::= \mathbf{X}\,\varphi_s \mid \mathbf{G}\,\varphi_s \mid \varphi_s\,\mathbf{U}\,\psi_s.$$

where p ranges over the set AP and A over the subsets of Agt.

In addition, we use standard abbreviations like $\top$, $\perp$, $\mathbf{F}$, etc. ATL formulae are interpreted over states of a game structure $\mathcal{S}$. The semantics of the main modalities is defined as follows[3]:

$$\begin{array}{rll}
\ell \models_{\mathcal{S}} \langle\!\langle A \rangle\!\rangle\, \varphi_p & \text{iff} & \exists F_A \in \mathsf{Strat}(A).\ \forall \rho \in \mathsf{Out}(\ell, F_A).\ \rho \models_{\mathcal{S}} \varphi_p, \\
\rho \models_{\mathcal{S}} \mathbf{X}\,\varphi_s & \text{iff} & \rho[1] \models_{\mathcal{S}} \varphi_s, \\
\rho \models_{\mathcal{S}} \mathbf{G}\,\varphi_s & \text{iff} & \forall i.\ \rho[i] \models_{\mathcal{S}} \varphi_s, \\
\rho \models_{\mathcal{S}} \varphi_s\,\mathbf{U}\,\psi_s & \text{iff} & \exists i.\ \rho[i] \models_{\mathcal{S}} \psi_s \text{ and } \forall 0 \le j < i.\ \rho[j] \models_{\mathcal{S}} \varphi_s.
\end{array}$$

It is well-known that, for the logic ATL, it is sufficient to restrict to state-based strategies (*i.e.*, $\langle\!\langle A \rangle\!\rangle\, \varphi_p$ is satisfied iff there is a state-based strategy all of whose outcomes satisfy φ_p) [3, 18].

Note that $\langle\!\langle \emptyset \rangle\!\rangle\, \varphi_p$ corresponds to the CTL formula $\mathbf{A}\varphi_p$ (*i.e.*, universal quantification over all computations issued from the current state), while $\langle\!\langle \mathsf{Agt} \rangle\!\rangle\, \varphi_p$

[2] *I.e.*, $f_{A_i}(\ell_0\cdots\ell_m) \in \mathsf{Mov}(\ell_m, A_i)$.

[3] We might omit to mention $\mathcal{S}$ when it is clear from the context.

corresponds to existential quantification $\mathbf{E}\varphi_p$. Note, however, that $\neg \langle\!\langle A\rangle\!\rangle\, \varphi_p$ is generally *not* equivalent to $\langle\!\langle \mathsf{Agt} \smallsetminus A\rangle\!\rangle\, \neg\varphi_p$ [3, 10]. Fig. 1 displays a (graphical representation of a) 2-player CGS for which, in ℓ_0, both $\neg \langle\!\langle A_1\rangle\!\rangle\, \mathbf{X}\, p$ and $\neg \langle\!\langle A_2\rangle\!\rangle\, \neg\mathbf{X}\, p$ hold. In such a representation, a transition is labeled with $\langle m_1.m_2\rangle$ when it correspond to move m_1 of player A_1 and to move m_2 of player A_2. Fig. 2 represents an (alternating-bisimilar) ATS with the same properties.

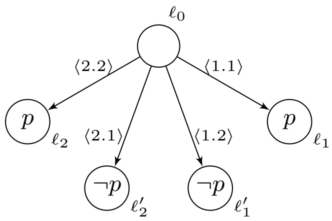


Fig. 1. A CGS that is not determined.

$$\mathsf{Loc} = \{\ell_0, \ell_1, \ell_2, \ell'_1, \ell'_2\}$$

$$\mathsf{Mov}(\ell_0, A_1) = \{\{\ell_1, \ell'_1\}, \{\ell_2, \ell'_2\}\}$$
$$\mathsf{Mov}(\ell_0, A_2) = \{\{\ell_1, \ell'_2\}, \{\ell_2, \ell'_1\}\}$$

$$\text{with} \begin{cases} \mathsf{Lab}(\ell_1) = \mathsf{Lab}(\ell_2) = \{p\} \\ \mathsf{Lab}(\ell'_1) = \mathsf{Lab}(\ell'_2) = \varnothing \end{cases}$$

Fig. 2. An ATS that is not determined.

Duality is a fundamental concept in modal and temporal logics: for instance, the dual of modality $\mathbf{U}$, often denoted by $\mathbf{R}$ and read *release*, is defined by $\varphi_s\, \mathbf{R}\, \psi_s \stackrel{\text{def}}{\equiv} \neg((\neg\varphi_s)\, \mathbf{U}\, (\neg\psi_s))$. Dual modalities allow, for instance, to put negations inner inside the formula, which is often an important property when manipulating formulas. In LTL, modality $\mathbf{R}$ can be expressed using only $\mathbf{U}$ and $\mathbf{G}$:

$$\varphi\, \mathbf{R}\, \psi \equiv \mathbf{G}\, \psi \vee \psi\, \mathbf{U}\, (\varphi \wedge \psi). \tag{1}$$

In the same way, it is well known that CTL can be defined using only modalities $\mathbf{EX}$, $\mathbf{EG}$ and $\mathbf{EU}$, and that we have

$$\mathbf{E}\varphi\, \mathbf{R}\, \psi \equiv \mathbf{EG}\, \psi \vee \mathbf{E}\psi\, \mathbf{U}\, (\varphi \wedge \psi) \qquad \mathbf{A}\varphi\, \mathbf{R}\, \psi \equiv \neg\, \mathbf{E}(\neg\varphi)\, \mathbf{U}\, (\neg\psi).$$

We prove in the sequel that modality $\mathbf{R}$ cannot be expressed in ATL, as defined in Definition 3. We thus define the following two extensions of ATL:

Definition 4. *We define $\mathsf{ATL}_{\mathbf{R}}$ and ATL^{+} with the following syntax:*

$$\begin{aligned} \mathsf{ATL}_{\mathbf{R}} \ni \varphi_s, \psi_s &::= \top \mid p \mid \neg\varphi_s \mid \varphi_s \vee \psi_s \mid \langle\!\langle A\rangle\!\rangle\, \varphi_p \\ \varphi_p &::= \mathbf{X}\, \varphi_s \mid \varphi_s\, \mathbf{U}\, \psi_s \mid \varphi_s\, \mathbf{R}\, \psi_s, \end{aligned}$$

$$\begin{aligned} \mathsf{ATL}^{+} \ni \varphi_s, \psi_s &::= \top \mid p \mid \neg\varphi_s \mid \varphi_s \vee \psi_s \mid \langle\!\langle A\rangle\!\rangle\, \varphi_p \\ \varphi_p, \psi_p &::= \neg\varphi_p \mid \varphi_p \vee \psi_p \mid \mathbf{X}\, \varphi_s \mid \varphi_s\, \mathbf{U}\, \psi_s \mid \varphi_s\, \mathbf{R}\, \psi_s. \end{aligned}$$

where p ranges over the set AP and A over the subsets of Agt.

Given a formula φ in one of the logics we have defined, the size of φ, denoted by $|\varphi|$, is the size of the tree representing that formula. The DAG-size of φ is the size of the directed acyclic graph representing that formula (*i.e.*, sharing common subformulas).

3 $\langle\!\langle A \rangle\!\rangle\,(a\,\mathrm{R}\,b)$ cannot be expressed in ATL

This section is devoted to the expressiveness of ATL. We prove:

Theorem 5. *There is no ATL formula equivalent to* $\Phi = \langle\!\langle A \rangle\!\rangle\,(a\,\mathbf{R}\,b)$.

The proof of Theorem 5 is based on techniques similar to those used for proving expressiveness results for temporal logics like CTL or ECTL [9]: we build two families of models $(s_i)_{i\in\mathbb{N}}$ and $(s'_i)_{i\in\mathbb{N}}$ s.t. (1) $s_i \not\models \Phi$, (2) $s'_i \models \Phi$ for any i, and (3) s_i and s'_i satisfy the same ATL formula of size less than i. Theorem 5 is a direct consequence of the existence of such families of models. In order to simplify the presentation, the theorem is proved for formula[4] $\Phi = \langle\!\langle A \rangle\!\rangle\,(b\,\mathbf{R}\,(a \vee b))$.

The models are described by one single inductive CGS [5] $\mathcal{C}$, involving two players. It is depicted on Fig. 3. A label $\langle \alpha.\beta \rangle$ on a transition indicates that this

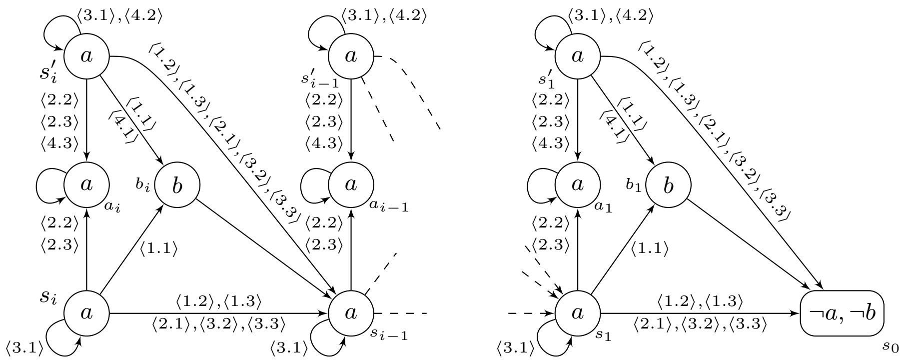


Fig. 3. The CGS $\mathcal{C}$, with states s_i and s'_i on the left

transition corresponds to move α of player A_1 and to move β of player A_2. In that CGS, states s_i and s'_i only differ in that player A_1 has a fourth possible move in s'_i. This ensures that, from state s'_i (for any i), player A_1 has a strategy (namely, he should always play 4) for enforcing $a\,\mathbf{W}\,b$. But this is not the case from state s_i: by induction on i, one can prove $s_i \not\models \langle\!\langle A_1 \rangle\!\rangle\, a\,\mathbf{W}\,b$. The base case is trivial. Now assume the property holds for i: from s_{i+1}, any strategy for A_1 starts with a move in $\{1, 2, 3\}$ and for any of these choices, player A_2 can choose a move (2, 1 and 2 resp.) that enforce the next state to be s_i where by i.h. A_1 has no strategy for $a\,\mathbf{W}\,b$.

We now prove that s_i and s'_i satisfy the same "small" formulae. First, we have the following equivalences:

[4] This formula can also be written $\langle\!\langle A \rangle\!\rangle\, a\,\mathbf{W}\,b$, where $\mathbf{W}$ is the "weak until" modality.

[5] Given the translation from CGS to ATS (see Section 4.2), the result also holds for ATSs.

Lemma 6. *For any $i > 0$, for any $\psi \in \mathit{ATL}$ with $|\psi| \le i$:*

$$b_i \models \psi \text{ iff } b_{i+1} \models \psi \qquad s_i \models \psi \text{ iff } s_{i+1} \models \psi \qquad s'_i \models \psi \text{ iff } s'_{i+1} \models \psi$$

The proof may be found in Appendix A.

Lemma 7. *$\forall i > 0$, $\forall \psi \in \mathit{ATL}$ with $|\psi| \le i$: $s_i \models \psi$ iff $s'_i \models \psi$.*

Proof. The proof proceeds by induction on i, and on the structure of the formula ψ. The case $i = 1$ is trivial, since s_1 and s'_1 carry the same atomic propositions. For the induction step, dealing with CTL modalities ($\langle\langle\emptyset\rangle\rangle$ and $\langle\langle A_1, A_2\rangle\rangle$) is also straightforward, then we just consider $\langle\langle A_1\rangle\rangle$- and $\langle\langle A_2\rangle\rangle$ modalities.

First we consider $\langle\langle A_1\rangle\rangle$-modalities. It is well-known that we can restrict to state-based strategies in this setting. If player A_1 has a strategy in s_i to enforce something, then he can follow the same strategy from s'_i. Conversely, if player A_1 has a strategy in s'_i to enforce some property, two cases may arise: either the strategy consists in playing move 1, 2 or 3, and it can be mimicked from s_i. Or the strategy consists in playing move 4 and we distinguish three cases:

- $\psi = \langle\langle A_1\rangle\rangle\, \mathbf{X}\, \psi_1$: that move 4 is a winning strategy entails that s'_i, a_i and b_i must satisfy ψ_1. Then s_i (by i.h. on the formula) and s_{i-1} (by Lemma 6) both satisfy ψ_1. Playing move 1 (or 3) in s_i ensures that the next state will satisfy ψ_1.
- $\psi = \langle\langle A_1\rangle\rangle\, \mathbf{G}\, \psi_1$: by playing move 4, the game could end up in s_{i-1} (*via* b_i), and in a_i and s'_i. Thus $s_{i-1} \models \psi$, and in particular ψ_1. By i.h., $s_i \models \psi_1$, and playing move 1 (or 3) in s_i, and then mimicking the original strategy (from s'_i), enforces $\mathbf{G}\, \psi_1$.
- $\psi = \langle\langle A_1\rangle\rangle\, \psi_1\, \mathbf{U}\, \psi_2$: a strategy starting with move 4 implies $s'_i \models \psi_2$ (the game could stay in s'_i for ever). Then $s_i \models \psi_2$ by i.h., and the result follows.

We now turn to $\langle\langle A_2\rangle\rangle$-modalities: clearly if $\langle\langle A_2\rangle\rangle\, \psi_1$ holds in s'_i, it also holds in s_i. Conversely, if player A_2 has a (state-based) strategy to enforce some property in s_i: If it consists in playing moves 1 or 3, then the same strategy also works in s'_i. Now if the strategy starts with move 2, then playing move 3 in s'_i has the same effect, and thus enforces the same property. □

Remark 1. ATL and $\mathsf{ATL}_{\mathbf{R}}$ have the same distinguishing power as the fragment of ATL involving only the $\langle\langle \cdot \rangle\rangle\, \mathbf{X}$ modality (see [4, proof of Th. 6]). This means that we cannot exhibit two models M and M' s.t. (1) $M \models \Phi$, (2) $M' \not\models \Phi$, and (3) M and M' satisfy the same ATL formula.

Even if ATL^+ would not contain the "release" modality in its syntax, it can express it, and it is thus strictly more expressive than ATL. However, as for CTL and CTL^+, it is possible to translate ATL^+ into $\mathsf{ATL}_{\mathbf{R}}$ [11]. Of course, such a translation induces at least an exponential blow-up in the size of the formulae since it is already the case when translating CTL^+ into CTL [21, 1]. Finally note that the standard model-checking algorithm for ATL easily extends to $\mathsf{ATL}_{\mathbf{R}}$ (and that MOCHA [5] handles $\mathsf{ATL}_{\mathbf{R}}$ formulae). In the same way, the axiomatization and satisfiability results of [10] can be extended to $\mathsf{ATL}_{\mathbf{R}}$ (as mentioned in the conclusion of [10]).

Turn-based games. In [3], a restriction of CGS —the turn-based CGSs— is considered. In any location of these models (named TB-CGS hereafter), only one player has several moves (the other players have only one possible choice). Such models have the property of *determinedness*: given a set of players A, either there is a strategy for A to win some objective Φ, or there is a strategy for other players ($\mathsf{Agt}\backslash A$) to enforce $\neg\Phi$. In such systems, modality $\mathbf{R}$ can be expressed as follows: $\langle\!\langle A\rangle\!\rangle\, \varphi\, \mathbf{R}\, \psi \equiv_{\text{TB-CGS}} \neg \langle\!\langle \mathsf{Agt}\backslash A\rangle\!\rangle\, (\neg\varphi)\, \mathbf{U}\, (\neg\psi)$.

4 Complexity of ATL model-checking

In this section, we establish the precise complexity of ATL model-checking. All the complexity results below are stated for ATL but they are also true for $\mathsf{ATL}_{\mathbf{R}}$.

Model-checking issues have been addressed in the seminal papers about ATL, on both ATSs [2] and CGSs [3]. The time complexity is shown to be in $O(m \cdot l)$, where m is the size of the transition table and l is the size of the formula. The authors then claim that the model-checking problem is in PTIME (and obviously, PTIME-complete). However, it is well-known (and already explained in [2, 3]) that the size m of the transition table may be exponential in the number of agents. Thus, when the transition table is not given explicitly (as is the case for ATS), the algorithm requires in fact exponential time.

Before proving that this problem is indeed not in PTIME, we define the model of *implicit* CGSs, with a succinct representation of the transition table [12]. Besides the theoretical aspect, it may be quite useful in practice since it allows to not explicitly describe the full transition table.

4.1 Explicit- and implicit CGSs

We distinguish between two classes of CGSs:

Definition 8. • *An* explicit CGS *is a CGS where the transition table is defined explicitly.*
• *An* implicit CGS *is a CGS where, in each location ℓ, the transition function is defined by a finite sequence $((\varphi_0, \ell_0), ..., (\varphi_n, \ell_n))$, where $\ell_i \in \mathsf{Loc}$ is a location, and φ_i is a boolean combination of propositions $A_j = c$ that evaluate to true iff agent A_i chooses move c. The transition table is then defined as follows: $\mathsf{Edg}(\ell, m_{A_1}, ..., m_{A_k}) = \ell_j$ iff j is the lowest index s.t. φ_j evaluates to true when players A_1 to A_k choose moves m_{A_1} to m_{A_k}. We require that the last boolean formula φ_i be $\top$, so that no agent can enforce a deadlock.*

The size $|\mathcal{C}|$ of a CGS $\mathcal{C}$ is defined as $|\mathsf{Loc}| + |\mathsf{Edg}|$. For explicit CGSs, $|\mathsf{Edg}|$ is the size of the transition table. For implicit CGSs, $|\mathsf{Edg}|$ is the sum $\sum |\varphi|$ used for the definition of Edg. See Appendix B for a discussion on the succinctness of the different models.

The size of an ATS is $|\mathsf{Loc}| + |\mathsf{Mov}|$ where $|\mathsf{Mov}|$ is the sum of the number of locations in each possible move of each agent in each location.

4.2 Expressiveness of CGSs and ATSs

We prove in this section that CGSs and ATSs are closely related: they can model the same concurrent games. In order to make this statement formal, we use the following definition:

Definition 9 ([4]). *Let $\mathcal{A}$ and $\mathcal{B}$ be two models of concurrent games (either ATSs or CGSs) over the same set Agt of agents. Let $R \subseteq \mathsf{Loc}_{\mathcal{A}} \times \mathsf{Loc}_{\mathcal{B}}$ be a (non-empty) relation between states of $\mathcal{A}$ and states of $\mathcal{B}$. That relation is an* alternating bisimulation *when, for any $(\ell, \ell') \in R$, the following conditions hold:*

- $\mathsf{Lab}_{\mathcal{A}}(\ell) = \mathsf{Lab}_{\mathcal{B}}(\ell')$;
- *for any coalition $A \subseteq \mathsf{Agt}$, we have*

$$\forall m \colon A \to \mathsf{Mov}_{\mathcal{A}}(\ell, A).\ \exists m' \colon A \to \mathsf{Mov}_{\mathcal{B}}(\ell', A).\\ \forall q' \in \mathsf{Next}(\ell', A, m').\ \exists q \in \mathsf{Next}(\ell, A, m).\ (q, q') \in R.$$

- *symmetrically, for any coalition $A \subseteq \mathsf{Agt}$, we have*

$$\forall m' \colon A \to \mathsf{Mov}_{\mathcal{B}}(\ell', A).\ \exists m \colon A \to \mathsf{Mov}_{\mathcal{A}}(\ell, A).\\ \forall q \in \mathsf{Next}(\ell, A, m).\ \exists q' \in \mathsf{Next}(\ell', A, m').\ (q, q') \in R.$$

where $\mathsf{Next}(\ell, A, m)$ is the set of locations that are reachable from ℓ when each player $A_i \in A$ plays $m(A_i)$.

Two models are said to be alternating-bisimilar if there exists an alternating bisimulation involving all of their locations.

With this equivalence in mind, ATSs and CGSs (both implicit and explicit ones) have the same expressive power:

Theorem 10. *1. Any explicit CGS can be translated into an alternating-bisimilar implicit one in linear time; 2. Any implicit CGS can be translated into an alternating-bisimilar explicit one in exponential time; 3. Any explicit CGS can be translated into an alternating-bisimilar ATS in cubic time; 4. Any ATS can be translated into an alternating-bisimilar explicit CGS in exponential time; 5. Any implicit CGS can be translated into an alternating-bisimilar ATS in exponential time; 6. Any ATS can be translated into an alternating-bisimilar implicit CGS in quadratic time;*

Figure 4 summarizes those results. From our complexity results (and the assumption that the polynomial-time hierarchy does not collapse), the costs of the above translations is optimal. Those translations are detailled in Appendix B.

4.3 Model checking ATL on implicit CGSs.

Basically, the algorithm for model-checking ATL [2, 3] is similar to that for CTL: it consists in recursively computing fixpoints, based *e.g.* on the following equivalence:

$$\langle\!\langle A \rangle\!\rangle\, p\, \mathbf{U}\, q \equiv \mu Z.(q \vee (p \wedge \langle\!\langle A \rangle\!\rangle\, \mathbf{X}\, Z))$$

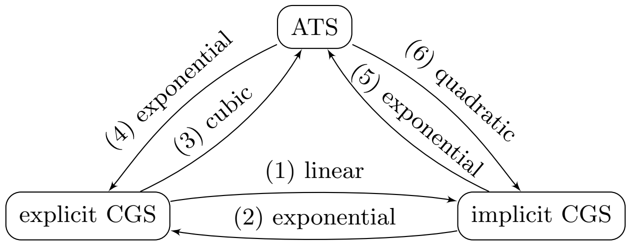


Fig. 4. Costs of translations between the three models

The difference with CTL is that we have to compute the pre-image of a set of states *for some coalition*.

It has been remarked in [12] that computing the pre-images is not in PTIME anymore when considering implicit CGSs: the algorithm has to non-deterministically guess the moves of players in A in each location, and for each pre-image, to solve the resulting SATqueries derived from those choices and from the transition table. As a consequence, model-checking ATL on implicit CGSs is Σ_2^{P}-hard [12]. However (see below), the Σ_2^{P}-hardness proof can very easily be adapted to prove Π_2^{P}-hardness. It follows that the Σ_2^{P} algorithm proposed in [12] cannot be correct. The flaw is in the way it handles negation: games played on CGSs (and ATSs) are generally not determined, and the fact that a player has no strategy to enforce φ does not imply that the other players have a strategy to enforce $\neg\varphi$. It rather means that the other players have a *co-strategy* to enforce $\neg\varphi$ (see [10] for precise explanations about co-strategies).

Still, the Σ_2^{P}-algorithm is correct for formulas whose main operator is not a negation. As a consequence:

Proposition 11. *Model checking ATL on implicit CGSs is in* Δ_3^{P}.

Since the algorithm consists in labeling the locations with the subformulae it satisfies, that complexity holds even if we consider the DAG-size of the formula.

Before proving optimality, we briefly recall the Σ_2^{P}-hardness proof of [12]. It relies on the following Σ_2^{P}-complete problem:

EQSAT$_2$:

Input: two families of variables $X = \{x^1, ..., x^n\}$ and $Y = \{y^1, ..., y^n\}$, a boolean formula φ on the set of variables $X \cup Y$.
Output: True iff $\exists X.\ \forall Y.\ \varphi$.

This problem can be encoded in an ATL model-checking problem on an implicit CGS: the CGS has three states q_1, $q_\top$ and $q_\perp$, and $2n$ agents A^1, ..., A^n, B^1, ..., B^n, each having two possible choices in q_1 and only one choice in $q_\top$ and $q_\perp$. The transitions out of $q_\top$ and $q_\perp$ are self loops. The transitions from q_1 are given by: $\delta(q_1) = ((\varphi[x^j \leftarrow (A^j \stackrel{?}{=} 1), y^j \leftarrow (B^j \stackrel{?}{=} 1)], q_\top)(\top, q_\perp))$.

Then clearly, the coalition A^1, ..., A^n has a strategy for reaching $q_\top$, *i.e.*, $q_1 \models \langle\!\langle A^1, ..., A^n \rangle\!\rangle \mathbf{X}\, q_\top$, iff there exists a valuation for variables in X s.t. φ is true whatever B-agents choose for Y.

Now, this encoding can easily be adapted to the dual (thus Π_2^{P}-complete) problem AQSAT$_2$, in which, with the same input, the output is the value of $\forall X.\ \exists Y.\ \varphi$. It suffices to consider the same implicit CGS, and the formula $\neg \langle\!\langle A^1, ..., A^n \rangle\!\rangle \mathbf{X} \neg q_\top$. It states that there is no strategy for players A^1 to A^n to avoid $q_\top$: whatever their choice, players B^1 to B^n can enforce φ.

Following the same idea, we prove the following result:

Proposition 12. *Model checking* ATL *on implicit CGSs is* Δ_3^{P}*-hard.*

Proof. We consider the following Δ_3^{P}-complete problem[14, 18].

SNSAT$_2$:

Input: m families of variables $X_i = \{x_i^1, ..., x_i^n\}$, m families of variables $Y_i = \{y_i^1, ..., y_i^n\}$, m variables z_i, m boolean formulae φ_i, with φ_i involving variables in $X_i \cup Y_i \cup \{z_1, ..., z_{i-1}\}$.

Output: The value of z_m, defined by

$$\begin{cases} z_1 \stackrel{\text{def}}{=} \exists X_1.\ \forall Y_1.\ \varphi_1(X_1, Y_1) \\ z_2 \stackrel{\text{def}}{=} \exists X_2.\ \forall Y_2.\ \varphi_2(z_1, X_2, Y_2) \\ \qquad \cdots \\ z_m \stackrel{\text{def}}{=} \exists X_m.\ \forall Y_m.\ \varphi_m(z_1, ..., z_{m-1}, X_m, Y_m) \end{cases}$$

We pick an instance $\mathcal{I}$ of this problem, and reduce it to an instance of the ATL model-checking problem. Note that such an instance uniquely defines the values of variables z_i. We write $v_{\mathcal{I}} \colon \{z_1, ..., z_m\} \to \{\top, \bot\}$ for this valuation. Also, when $v_{\mathcal{I}}(z_i) = \top$, there exists a witnessing valuation for variables in X_i. We extend $v_{\mathcal{I}}$ to $\{z_1, ..., z_m\} \cup \bigcup_i X_i$, with $v_{\mathcal{I}}(x_i^j)$ being a witnessing valuation if $v_{\mathcal{I}}(z_i) = \top$.

We now define an implicit CGS $\mathcal{C}$ as follows: it contains mn agents A_i^j (one for each x_i^j), mn agents B_i^j (one for each y_i^j), m agents C_i (one for each z_i), and one extra agent D. The structure is made of m states q_i, m states $\overline{q_i}$, m states s_i, and two states $q_\top$ and $q_\bot$. There are three atomic propositions: $s_\top$ and $s_\bot$, that label the states $q_\top$ and $q_\bot$ resp., and an atomic proposition s labeling states s_i. The other states carry no label.

Except for D, the agents represent booleans, and thus always have two possible choices (0 and 1). Agent D always has m choices (0 to $m-1$). The transition relation is defined as follows: for each i,

$$\begin{aligned} \delta(\overline{q_i}) &= ((\top, s_i)); \\ \delta(s_i) &= ((\top, q_i)); \\ \delta(q_\top) &= ((\top, q_\top)); \\ \delta(q_\bot) &= ((\top, q_\bot)); \end{aligned} \qquad \delta(q_i) = \begin{pmatrix} ((D \stackrel{?}{=} 0) \wedge \varphi_i[x_i^j \leftarrow (A_i^j \stackrel{?}{=} 1), \\ \qquad y_i^j \leftarrow (B_i^j \stackrel{?}{=} 1), z_k \leftarrow (C_k \stackrel{?}{=} 1)], q_\top) \\ ((D \stackrel{?}{=} 0), q_\bot) \\ ((D \stackrel{?}{=} k) \wedge (C_k \stackrel{?}{=} 1), q_k) \text{ for each } k < i \\ ((D \stackrel{?}{=} k) \wedge (C_k \stackrel{?}{=} 0), \overline{q_k}) \text{ for each } k < i \\ (\top, q_\top) \end{pmatrix}$$

Intuitively, from state q_i, the boolean agents chose a valuation for the variable they represent, and agent D can either choose to check if the valuation really witnesses φ_i (by choosing move 0), or "challenge" player C_k, with move $k < i$.

The ATL formula is built recursively by $\psi_0 = \top$ and, writing AC for the coalition $\{A_1^1, ..., A_m^n, C_1, ..., C_m\}$: $\psi_{k+1} \stackrel{\text{def}}{=} \langle\!\langle \mathsf{AC} \rangle\!\rangle\, (\neg s)\, \mathbf{U}\, (q_\top \vee \mathbf{EX}\, (s \wedge \mathbf{EX}\, \neg \psi_k))$.

Let $f_\mathcal{I}(A)$ be the state-based strategy for agent $A \in \mathsf{AC}$ that consists in playing according to the valuation $v_\mathcal{I}$ (*i.e.* move 0 if the variable associated with A evaluates to 0 in $v_\mathcal{I}$, and move 1 otherwise). The following lemma (proved in Appendix C) completes the proof of Proposition 12:

Lemma 13. *For any $i \leq m$ and $k \geq i$, the following three statements are equivalent: (a) $\mathcal{C}, q_i \models \psi_k$; (b) the strategies $f_\mathcal{I}$ witness the fact that $\mathcal{C}, q_i \models \psi_k$; (c) variable z_i evaluates to $\top$ in $v_\mathcal{I}$.* □

With Proposition 11, this implies:

Theorem 14. *Model checking ATL on implicit CGSs is Δ_3^P-complete.*

4.4 Model checking ATL on ATSs.

For ATSs also, the PTIME upper bound only holds when the number of agents is fixed. As in the previous section, the NP algorithm proposed in [12] for ATL model-checking on ATSs does not handle negation correctly. Again, the algorithm consists in computing fixpoints with pre-images, and the pre-images are now computed in NP [12]. This yields a Δ_2^P algorithm for full ATL.

Proposition 15. *Model checking ATL over ATSs is in Δ_2^P.*

The NP-hardness proof of [12] can be adapted in order to give a direct reduction of 3SAT, and then extended to SNSAT:

Proposition 16. *Model checking ATL on ATSs is Δ_2^P-hard.*

Proof. Let us first recall the definition of the SNSAT problem [14]:

SNSAT:

Input: p families of variables $X_r = \{x_r^1, ..., x_r^m\}$, p variables z_r, p boolean formulae φ_r in 3-CNF, with φ_r involving variables in $X_r \cup \{z_1, ..., z_{r-1}\}$.

Output: The value of z_p, defined by

$$\begin{cases} z_1 \stackrel{\text{def}}{=} \exists X_1.\ \varphi_1(X_1) \\ z_2 \stackrel{\text{def}}{=} \exists X_2.\ \varphi_2(z_1, X_2) \\ z_3 \stackrel{\text{def}}{=} \exists X_3.\ \varphi_3(z_1, , z_2, X_3) \\ \qquad \ldots \\ z_p \stackrel{\text{def}}{=} \exists X_p.\ \varphi_p(z_1, ..., z_{p-1}, X_p) \end{cases}$$

Let $\mathcal{I}$ be an instance of SNSAT, where we assume that each φ_r is made of n clauses S_r^1 to S_r^n, with $S_r^j = \alpha_r^{j,1} s_r^{j,1} \vee \alpha_r^{j,2} s_r^{j,2} \vee \alpha_r^{j,3} s_r^{j,3}$. Again, such an instance uniquely defines a valuation $v_{\mathcal{I}}$ for variables z_1 to z_r, that can be extended to the whole set of variables by choosing a witnessing valuation for x_r^1 to x_r^n when z_r evaluates to true.

We now describe the ATS $\mathcal{A}$: it contains $(8n+3)p$ states: p states $\overline{q_r}$ and p states q_r, p states s_r, and for each formula φ_r, for each clause S_r^j of φ_r, eight states $q_r^{j,0}, ..., q_r^{j,7}$, as in the previous reduction.

States s_r are labelled with the atomic proposition s, and states $q_r^{j,k}$ that do not correspond to clause S_r^j are labeled with α.

There is one player A_r^j for each variable x_r^j, one player C_r for each z_r, plus one extra player D. As regards transitions, there are self-loops on each state $q_r^{j,k}$, single transitions from each $\overline{q_r}$ to the corresponding s_r, and from each s_r to the corresponding q_r. From state q_r,

- player A_r^j will choose the value of variable x_r^j, by selecting one of the following two sets of states:

$$\{q_r^{g,k} \mid \forall l \le 3.\ s_r^{g,l} \neq x_r^j \text{ or } \alpha_r^{g,l} = 0\} \cup \{q_t, \overline{q_t} \mid t < r\} \qquad \text{if } x_r^j = \top$$
$$\{q_r^{g,k} \mid \forall l \le 3.\ s_r^{g,l} \neq x_r^j \text{ or } \alpha_r^{g,l} = 1\} \cup \{q_t, \overline{q_t} \mid t < r\} \qquad \text{if } x_r^j = \bot$$

  Both choices also allow to go to one of the states q_t or $\overline{q_t}$. In q_r, players A_t^j with $t \neq r$ have one single choice, which is the whole set of states.
- player C_t also chooses for the value of the variable it represents. As for players A_r^j, this choice will be expressed by choosing between two sets of states corresponding to clauses that are not made true. But as in the proof of Prop. 12, players C_t will also offer the possibility to "verify" their choice, by going either to state q_t or $\overline{q_t}$. Formally, this yields two sets of states:

$$\{q_r^{g,k} \mid \forall l \le 3.\ s_r^{g,l} \neq z_t \text{ or } \alpha_r^{g,l} = 0\} \cup \{q_u, \overline{q_u} \mid u \neq t\} \cup \{q_t\} \qquad \text{if } z_t = \top$$
$$\{q_r^{g,k} \mid \forall l \le 3.\ s_r^{g,l} \neq z_t \text{ or } \alpha_r^{g,l} = 1\} \cup \{q_u, \overline{q_u} \mid u \neq t\} \cup \{\overline{q_t}\} \qquad \text{if } z_t = \bot$$

- Last, player D chooses either to challenge a player C_t, with $t < r$, by choosing the set $\{q_t, \overline{q_t}\}$, or to check that a clause S_r^j is fulfilled, by choosing $\{q_r^{j,0}, ..., q_r^{j,7}\}$.

Let us first prove that any choices of all the players yields exactly one state. It is obvious except for states q_r. For a state q_r, let us first restrict to the choices of all the players A_r^j and C_r, then:

- if we only consider states $q_r^{1,0}$ to $q_r^{n,7}$, the same argument as in the previous proof ensures that precisely on state per clause is chosen,
- if we consider states q_t and $\overline{q_t}$, the choices of players B_t ensure that exactly one state has been chosen in each pair $\{q_t, \overline{q_t}\}$, for each $t < r$.

Clearly, the choice of player D will select exactly one of the remaining states.

Now, we build the ATL formula. It is a recursive formula (very similar to the one used in the proof of Prop. 12), defined by $\psi_0 = \top$ and (again writing AC for the set of players $\{A_1^1, ..., A_p^m, C_1, ..., C_p\}$):

$$\psi_{r+1} \stackrel{\text{def}}{=} \langle\!\langle \mathsf{AC} \rangle\!\rangle\, (\neg s)\, \mathbf{U}\, (\alpha \vee \mathbf{EX}\, (s \wedge \mathbf{EX}\, \neg\psi_r)).$$

Then, writing $f_{\mathcal{I}}$ for the state-based strategy associated to $v_{\mathcal{I}}$:

Lemma 17. *For any $r \leq p$ and $t \geq r$, the following statements are equivalent: (a) $q_r \models \psi_t$; (b) the strategies $f_{\mathcal{I}}$ witness the fact that $q_r \models \psi_t$; (c) variable z_r evaluates to true in $v_{\mathcal{I}}$.*

The technical proof of this lemma is given in Appendix D. In the end:

Theorem 18. *Model checking ATL on ATSs is Δ_2^{P}-complete.*

4.5 Model checking ATL^+

The complexity of model checking ATL^+ over ATSs has been settled Δ_3^{P}-complete in [18]. But Δ_3^{P}-hardness proof of [18] is in $\mathsf{LOGSPACE}$ only w.r.t. the DAG-size of the formula. We prove (in Appendix E) that model checking ATL^+ is in fact Δ_3^{P}-complete (with the standard definition of the size of a formula) for our three kinds of game structures.

Theorem 19. *Model checking ATL^+ is Δ_3^{P}-complete on ATSs as well as on explicit CGSs and implicit CGSs.*

5 Conclusion

In this paper, we considered the basic questions of expressiveness and complexity of ATL. We showed that ATL, as originaly defined in [2, 3], is not as expressive as it could be expected, and we argue that the modality "Release" should be added in its definition [13].

We also precisely characterized the complexity of ATL and ATL^+ model-checking, on both ATSs and CGSs, when the number of agents is not fixed. These results complete the previously known results about these formalisms and it is interesting to see that their complexity classes (Δ_2^{P} or Δ_3^{P}) are unusual in the model-checking area.

As future works, we plan to focus on the extensions EATL (extending ATL with a modality $\langle\!\langle \cdot \rangle\!\rangle \overset{\infty}{\mathbf{F}}$, and for which state-based strategies are still sufficient) and EATL^+ (the obvious association of both extensions, but for which state-based strategies are not sufficient anymore).

Acknowledgement. We thank Wojtek Jamroga for pointing out that formulas in SNSAT_2 cannot be restricted to CNF [6].

References

1. M. Adler and N. Immerman. An $n!$ lower bound on formula size. In *Proceedings of the 16th Annual Symposium on Logic in Computer Science (LICS'01)*, p. 197–206. IEEE CSP, 2001.

2. R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time temporal logic. In *Proc. 38th Annual Symp. Foundations of Computer Science (FOCS'97)*, p. 100–109. IEEE CSP, 1997.
3. R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *J. ACM*, 49(5):672–713, 2002.
4. R. Alur, T. A. Henzinger, O. Kupferman, and M. Y. Vardi. Alternating refinement relations. In *Proc. 9th Intl Conf. Concurrency Theory (CONCUR'98)*, vol. 1466 of *LNCS*, p. 163–178. Springer, 1998.
5. R. Alur, T. A. Henzinger, F. Y. C. Mang, S. Qadeer, S. K. Rajamani, and S. Tasiran. Mocha: Modularity in model checking. In *Proc. 10th Intl Conf. Computer Aided Verification (CAV'98)*, vol. 1427 of *LNCS*, p. 521–525. Springer, 1998.
6. M. Cadoli, A. Giovanardi, and M. Schaerf. An algorithm to evaluate quantified boolean formulae. *Journal of Automated Reasoning*, 28(2):101–142, 2002.
7. E. M. Clarke and E. A. Emerson. Design and synthesis of synchronous skeletons using branching-time temporal logic. In *Proc. 3rd Workshop Logics of Programs (LOP'81)*, vol. 131 of *LNCS*, p. 52–71. Springer, 1981.
8. E. M. Clarke, E. A. Emerson, and A. P. Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Trans. on Prog. Lang. and Systems (TOPLAS)*, 8(2):244–263, 1986.
9. E. A. Emerson. Temporal and modal logic. In *Handbook of Theoretical Computer Science*, vol. B, chapter 16, p. 995–1072. Elsevier, 1990.
10. V. Goranko and G. van Drimmelen. Complete axiomatization and decidability of alternating-time temporal logic. *Theoretical Computer Science*, 353(1-3):93–117, Mar. 2006.
11. A. Harding, M. Ryan, and P.-Y. Schobbens. Approximating ATL* in ATL. In *Revised Papers 3rd Intl Workshop Verification, Model Checking, and Abstract Interpretation (VMCAI'02)*, vol. 2294 of *LNCS*, p. 289–301. Springer, 2002.
12. W. Jamroga and J. Dix. Do agents make model checking explode (computationally)? In *Proc. 4th Intl Centr. and East. Europ. Conf. Multi-Agent Systems (CEEMAS'05)*, vol. 3690 of *LNCS*. Springer, 2005.
13. W. Jamroga and J. Dix. Model checking abilities of agents: A closer look. Technical Report IfI-06-02, Institut für Informatik, Technische Universität Clausthal, 2006.
14. F. Laroussinie, N. Markey, and Ph. Schnoebelen. Model checking CTL$^+$ and FCTL is hard. In *Proc. 4th Intl Conf. Foundations of Software Science and Computation Structure (FoSSaCS'01)*, vol. 2030 of *LNCS*, p. 318–331. Springer, 2001.
15. Ch. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.
16. A. Pnueli. The temporal logic of programs. In *Proc. 18th Ann. Symp. Foundations of Computer Science (FOCS'77)*, p. 46–57. IEEE Comp. Soc. Press, 1977.
17. J.-P. Queille and J. Sifakis. Specification and verification of concurrent systems in CESAR. In *Proc. 5th Intl Symp. on Programming (SOP'82)*, vol. 137 of *LNCS*, p. 337–351. Springer, 1982.
18. P.-Y. Schobbens. Alternating-time logic with imperfect recall. In *Proc. 1st Workshop Logic and Communication in Multi-Agent Systems (LCMAS'03)*, vol. 85 of *ENTCS*. Elsevier, 2004.
19. D. Walther, C. Lutz, F. Wolter, and M. Wooldridge. ATL satisfiability is indeed EXPTIME-complete. *Journal of Logic and Computation*, 2006. To appear.
20. I. Walukiewicz. A landscape with games in the background. In *Proc. 19th Ann. Symp. Logic in Computer Science (LICS'04)*, p. 356–366. IEEE CSP, 2004.
21. Th. Wilke. CTL$^+$ is exponentially more succinct than CTL. In *Proc. 19th Conf. Foundations of Software Technology and Theoretical Computer Science (FSTTCS'99)*, vol. 1738 of *LNCS*, p. 110–121. Springer, 1999.

A Proof of Lemma 6

Lemma 6. *Consider the CGS $\mathcal{C}$ displayed at Fig. 3. For any $i > 0$, for any $\psi \in \mathsf{ATL}$ with $|\psi| \leq i$,*

$$b_i \models \psi \text{ iff } b_{i+1} \models \psi \tag{2}$$

$$s_i \models \psi \text{ iff } s_{i+1} \models \psi \tag{3}$$

$$s'_i \models \psi \text{ iff } s'_{i+1} \models \psi \tag{4}$$

Proof. The proof proceeds by induction on i, and on the structure of the formula ψ.

Base case: $i = 1$. Since we require that $|\psi| \leq i$, ψ can only be an atomic proposition. The result is then obvious.

Induction step. We assume the result holds up to some $i - 1 \geq 1$, and prove that it then still holds for i. Let ψ s.t. $|\psi| \leq i$. We now proceed by structural induction on ψ:

- The result is again obvious for atomic propositions, as well as for boolean combinations of subformulae.
- Otherwise, the "root" combinator of ψ is a modality. If it is a CTL modality, the results are quite straightforward. Also, since there is only one transition from b_i, any ATL modality can be expressed as a CTL modality in that state, and (2) follows.
- If $\psi = \langle\!\langle A_1 \rangle\!\rangle \mathbf{X}\, \psi_1$: Assume $s_i \models \psi$. Then, depending on the strategy, either b_i and s_{i-1}, or a_i and s_{i-1}, or s_i and s_{i-1}, should satisfy ψ_1. By i.h., this propagates to the next level, and the same strategy can be mimicked from s_{i+1}.
  The converse is similar (hence (3)), as well as the proof for (4).
- If $\psi = \langle\!\langle A_1 \rangle\!\rangle \mathbf{G}\, \psi_1$: If $s_i \models \psi$, then s_i, thus s_{i+1}, satisfy ψ_1. Playing move 3 is a strategy for player A_1 to enforce $\mathbf{G}\, \psi_1$ from s_{i+1}, since the game will either stay in s_{i+1} or go to s_i, where player A has a winning strategy.
  The converse is immediate, since player A_1 cannot avoid s_i when playing from s_{i+1}. Hence (3) for $\langle\!\langle A_1 \rangle\!\rangle \mathbf{G}$-formulae.

  If $s'_i \models \psi$, then both s'_i and s'_{i+1} satisfy ψ_1. Also, player A_1 cannot avoid the play to go in location s_{i-1}. Thus, $s_{i-1} \models \psi_1$ —and by i.h., so does s_i— and $s_i \models \psi$, as above. Now, following the same strategy in s'_{i+1} as the winning strategy of s'_i clearly enforces $\mathbf{G}\, \psi_1$. The converse is similar: it suffices to mimic, from s'_i, the strategy witnessing the fact that $s'_{i+1} \models \psi$. This proves (4), and concludes this case.
- If $\psi = \langle\!\langle A_1 \rangle\!\rangle \psi_1 \mathbf{U}\, \psi_2$: If $s_i \models \psi$, then either ψ_2 or ψ_1 holds in s_i, thus in s_{i+1}. The former case is trivial. In the latter, player A_1 can mimic the winning strategy in s_{i+1}: the game will end up in s_i, with intermediary states satisfying ψ_1 (or ψ_2), and he can then apply the original strategy.

The converse is obvious, since from s_{i+1}, player A_1 cannot avoid location s_i, from which he must also have a winning strategy.

If $s'_i \models \psi$, omitting the trivial case where s'_i satisfies ψ_2, we have that $s_{i-1} \models \psi$. Also, a (state-based) strategy in s'_i witnessing ψ necessary consists in playing move 1 or 2. Thus a_i and b_i satisfy ψ, and the same strategy (move 1 or 2, resp.) enforces $\mathbf{G}\,\psi_1$ from s_i. It is now easy to see that the same strategy is correct from s'_{i+1}.
Conversely, apart from trivial cases, the strategy can again only consists in playing moves 1 or 2. In both cases, the game could end up in s_i, and then in s_{i-1}. Thus $s_{i-1} \models \psi$, and the same strategy as in s'_{i+1} can be applied in s'_i to witness ψ.
- The proofs for $\langle\!\langle A_2 \rangle\!\rangle\, \mathbf{X}\, \psi_1$, $\langle\!\langle A_2 \rangle\!\rangle\, \mathbf{X}\, \psi_1$, and $\langle\!\langle A_2 \rangle\!\rangle\, \psi_1 \,\mathbf{U}\, \psi_2$ are very similar to the previous ones. □

B From ATSs to CGSs

Theorem 10. *1. An explicit CGS can be translated into an alternating-bisimilar implicit one in linear time;*
2. An implicit CGS can be translated into an alternating-bisimilar explicit one in exponential time;
3. An explicit CGS can be translated into an alternating-bisimilar ATS in cubic time;
4. An ATS can be translated into an alternating-bisimilar explicit CGS in exponential time;
5. An implicit CGS can be translated into an alternating-bisimilar ATS in exponential time;
6. An ATS can be translated into an alternating-bisimilar implicit CGS in quadratic time;

Proof. Points 1, 2, and 4 are reasonnably easy.

For point 6, it suffices to write, for each possible next location, the conjunction (on each agent) of the disjunction of the choices that contain that next location. For instance, if we have $\mathsf{Mov}_\mathcal{A}(\ell_0, A_1) = \{\{\ell_1, \ell_2\}, \{\ell_1, \ell_3\}\}$ and $\mathsf{Mov}_\mathcal{A}(\ell_0, A_2) = \{\{\ell_2, \ell_3\}, \{\ell_1\}\}$ in the ATS $\mathcal{A}$, then each player will have two choices in the associated CGS $\mathcal{B}$, and

$$\mathsf{Edg}_\mathcal{B}(\ell_0) = \begin{pmatrix} (A_1 = 1 \vee A_1 = 2) \wedge (A_2 = 2), \ell_1 \\ (A_1 = 1) \wedge (A_2 = 1), \ell_2 \\ (A_1 = 2) \wedge (A_2 = 1), \ell_3 \end{pmatrix}$$

Formally, let $\mathcal{A} = (\mathsf{Agt}, \mathsf{Loc}_\mathcal{A}, \mathsf{AP}, \mathsf{Lab}_\mathcal{A}, \mathsf{Mov}_\mathcal{A})$ be an ATS. We then define $\mathcal{B} = (\mathsf{Agt}, \mathsf{Loc}_\mathcal{B}, \mathsf{AP}, \mathsf{Lab}_\mathcal{B}, \mathsf{Mov}_\mathcal{B}, \mathsf{Edg}_\mathcal{B})$ as follows:

- $\mathsf{Loc}_\mathcal{B} = \mathsf{Loc}_\mathcal{A}$, $\mathsf{Lab}_\mathcal{B} = \mathsf{Lab}_\mathcal{A}$;
- $\mathsf{Mov}_\mathcal{B} \colon \ell \times A_i \to [1, |\mathsf{Mov}_\mathcal{A}(\ell, A_i)|]$;

– $\mathsf{Edg}_{\mathcal{B}}$ is a function mapping each location ℓ to the sequence $((\varphi_{\ell'}, \ell'))_{\ell' \in \mathsf{Loc}_{\mathcal{A}}}$ (the order is not important here, as the formulas will be mutually exclusive) with

$$\varphi_{\ell'} = \bigwedge_{A_i \in \mathsf{Agt}} \left(\bigvee_{\substack{\ell' \text{ appears in the } j\text{-th} \\ \text{set of } \mathsf{Mov}_{\mathcal{A}}(\ell, A_i)}} A_i \stackrel{?}{=} j \right)$$

Computing $\mathsf{Edg}_{\mathcal{B}}$ requires quadratic time (more precisely $O(|\mathsf{Loc}_{\mathcal{A}}| \times |\mathsf{Mov}_{\mathcal{A}}|)$). It is now easy to prove that the identity $\mathrm{Id} \subseteq \mathsf{Loc}_{\mathcal{A}} \times \mathsf{Loc}_{\mathcal{B}}$ is an alternating bisimulation, since there is a direct correspondance between the choices in both structures.

We now explain how to transform an explicit CGS into an ATS, showing point 3. Let $\mathcal{A} = (\mathsf{Agt}, \mathsf{Loc}_{\mathcal{A}}, \mathsf{AP}, \mathsf{Lab}_{\mathcal{A}}, \mathsf{Mov}_{\mathcal{A}}, \mathsf{Edg}_{\mathcal{A}})$ be an explicit CGS. We define the ATS $\mathcal{B} = (\mathsf{Agt}, \mathsf{Loc}_{\mathcal{B}}, \mathsf{AP}, \mathsf{Lab}_{\mathcal{B}}, \mathsf{Mov}_{\mathcal{B}})$ as follows (see Figure 5 for more intuition on the construction):

– $\mathsf{Loc}_{\mathcal{B}} \subseteq \mathsf{Loc}_{\mathcal{A}} \times \mathsf{Loc}_{\mathcal{A}} \times \mathbb{N}^k$, where $k = |\mathsf{Agt}|$, with $(\ell, \ell', m_{A_1}, \ldots, m_{A_k}) \in \mathsf{Loc}_{\mathcal{B}}$ iff $\ell = \mathsf{Edg}_{\mathcal{A}}(\ell', m_{A_1}, \ldots, m_{A_k})$;
– $\mathsf{Lab}_{\mathcal{B}}(\ell, \ell', m_{A_1}, \ldots, m_{A_k}) = \mathsf{Lab}_{\mathcal{A}}(\ell)$;
– From a location $q = (\ell, \ell', m_{A_1}, \ldots, m_{A_k})$, player A_j has $|\mathsf{Mov}_{\mathcal{A}}(\ell, A_j)|$ possible moves:

$$\begin{aligned}\mathsf{Mov}_{\mathcal{B}}(q, A_j) = \Big\{ \big\{ (\ell'', \ell, m'_{A_1}, \ldots, m'_{A_j} = i, \ldots, m'_{A_k}) \ \mid\ m'_{A_n} \in \mathsf{Mov}_{\mathcal{A}}(\ell, A_n) \\ \text{and } \ell'' = \mathsf{Edg}_{\mathcal{A}}(\ell, m_{A_1}, \ldots, m_{A_j} = i, \ldots, m_{A_k}) \big\} \ \mid\ i \in \mathsf{Mov}_{\mathcal{A}}(\ell, A_j) \Big\}\end{aligned}$$

This ATS is built in time $O(|\mathsf{Loc}_{\mathcal{A}}|^2 \cdot |\mathsf{Edg}_{\mathcal{A}}|)$. It remains to show alternating bisimilarity between those structures. We define the relation

$$R = \{(\ell, (\ell, \ell', m_{A_1}, \ldots, m_{A_k})) \ \mid\ \ell \in \mathsf{Loc}_{\mathcal{A}}, (\ell, \ell', m_{A_1}, \ldots, m_{A_k})) \in \mathsf{Loc}_{\mathcal{B}}\}.$$

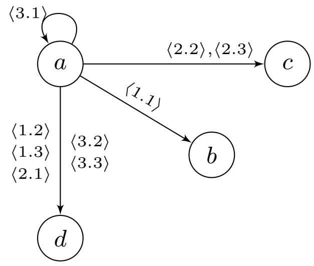
It is now only a matter of bravery to prove that R is an alternating bisimulation between $\mathcal{A}$ and $\mathcal{B}$.

Point 5 is now immediate (through explicit CGSs), but it could also be proved in a similar way as point 3. $\square$

Let us mention that our translations are optimal (up to a polynomial): our exponential translations cannot be achieved in polynomial time because of our complexity results for ATL model-checking. Note that it does not mean that the resulting structures must have exponential size.

C Proof of Lemma 13

Lemma 13. *For any $i \leq m$ and $k \geq i$, the following three statements are equivalent:*



Moves from location A:

Player 1
move 1: $\{b_{a,1,1}, d_{a,1,2}, d_{a,1,3}\}$
move 2: $\{c_{a,2,2}, c_{a,2,3}, d_{a,2,1}\}$
move 3: $\{a_{a,3,1}, d_{a,3,2}, d_{a,3,3}\}$

Player 2
move 1: $\{a_{a,3,1}, b_{a,1,1}, d_{a,2,1}\}$
move 2: $\{c_{a,2,2}, d_{a,1,2}, d_{a,3,2}\}$
move 3: $\{c_{a,2,3}, d_{a,1,3}, d_{a,3,3}\}$

Fig. 5. Converting an explicit CGS into an ATS

(a) $\mathcal{C}, q_i \models \psi_k$*;*
(b) the strategies $f_\mathcal{I}$ *witness the fact that* $\mathcal{C}, q_i \models \psi_k$*;*
(c) variable z_i *evaluates to true in* $v_\mathcal{I}$*.*

Proof. Clearly, *(b)* implies *(a)*. We prove that *(a)* implies *(c)* and that *(c)* implies *(b)* by induction on i.

First assume that $q_1 \models \psi_j$, for some $j \geq 1$. Since only $q_\top$ and $q_\bot$ are reachable from q_1, we have $q_1 \models \langle\!\langle \mathsf{AC} \rangle\!\rangle \mathbf{X}\, q_\top$. We are (almost) in the same case as in the Σ_2^P reduction of [12]: there is a valuation of the variables x_1^1 to x_1^n s.t., whatever players D and B_1^1 to B_m^n decide, the run will end up in $q_\top$. This holds in particular if player D chooses move 0: for any valuation of the variables y_1^1 to y_1^n, $\psi_1(X_1, Y_1)$ holds true, and z_1 evaluates to true in $v_\mathcal{I}$.

Secondly, if z_1 evaluates to true, then $v_\mathcal{I}(x_1^1)$, ..., $v_\mathcal{I}(x_1^n)$ are such that, whatever the value of y_1^1 to y_1^n, ψ_1 holds true. If players A_1^1 to A_1^n play according to $f_\mathcal{I}$, then players D and B_1^1 to B_1^n cannot avoid state $q_\top$, and $q_1 \models \langle\!\langle \mathsf{AC} \rangle\!\rangle \mathbf{X}\, q_\top$, thus also ψ_k when $k \geq 1$.

We now assume the result holds up to index $i \geq 1$, and prove that it also holds at step $i+1$. Assume $q_{i+1} \models \psi_{k+1}$, with $k \geq i$. There exists a strategy witnessing ψ_{k+1}, *i.e.*, s.t. all the outcomes following this strategy satisfy $(\neg s)\, \mathbf{U}\, (q_\top \vee \mathbf{EX}\, (s \wedge \mathbf{EX}\, \neg\psi_k))$. Depending on the move of player D in state q_{i+1}, we get several informations: first, if player D plays move l, with $1 \leq l \leq i$, the play goes to state q_l or $\overline{q_l}$, depending on the choice of player C_l.

- if player C_l chose move 0, the run ends up in $\overline{q_l}$. Since the only way out of that state is to enter state s_l, labeled by s, we get that $\overline{q_l} \models \mathbf{EX}\, (s \wedge \mathbf{EX}\, \neg\psi_k)$, *i.e.*, that $q_l \models \neg\psi_k$. By i.h., we get that z_l evaluates to false in our instance of SNSAT_2.
- if player C_l chose move 1, the run goes to q_l. In that state, players in AC can keep on applying their strategy, which ensures that $q_l \models \psi_{k+1}$, and, by i.h., that z_l evaluates to true in $\mathcal{I}$.

Thus, the strategy for AC to enforce ψ_{k+1} in q_{i+1} requires players C_1 to C_i to play according to $v_\mathcal{I}$ and the validity of these choices can be verified by the "opponent" D.

Now, if player D chooses move 0, all the possible outcomes will necessarily immediately go to $q_\top$ (since ψ_{k+1} holds, and since $q_\perp \not\models \mathbf{EX}\,(s \wedge \mathbf{EX}\,\neg\psi_k)$). We immediately get that players B^1_{i+1} to B^n_{i+1} cannot make ψ_{i+1} false, hence that z_{i+1} evaluates to true in $\mathcal{I}$.

Secondly, if z_{i+1} evaluates to true, assume players in AC play according to $f_\mathcal{I}$, and consider the possible moves of player D:

- if player D chooses move 0, since z_{i+1} evaluates to true and since players C_1 to C_i and A^1_{i+1} to A^n_{i+1} have played according $v_\mathcal{I}$, there is no way for player B^1_{i+1} to B^n_{i+1} to avoid state $q_\top$.
- if player D chooses some move l between 1 and i, the execution will go into state q_l or $\overline{q_l}$, depending on the move of C_l.
  - if C_l played move 0, *i.e.*, if z_l evaluates to false in $v_\mathcal{I}$, the execution goes to state $\overline{q_l}$, and we know by i.h. that $q_l \models \neg\psi_k$. Thus $\overline{q_l} \models \mathbf{EX}\,(s \wedge \mathbf{EX}\,\neg\psi_k)$, and the strategy still fulfills the requirement.
  - if C_l played move 1, *i.e.*, if z_l evaluates to true, then the execution ends up in state q_l, in which, by i.h., the strategy $f_\mathcal{I}$ enforces ψ_{k+1}.
- if player D plays some move l with $l > i$, the execution goes directly to $q_\top$, and the formula is fulfilled. □

D Proof of Lemma 17

Lemma 17. *For any $r \leq p$ and $t \geq r$, the following statements are equivalent:*

(a) $q_r \models \psi_t$*;*
(b) the strategies $f_\mathcal{I}$ witness the fact that $q_r \models \psi_t$;
(c) variable z_r evaluates to true in $v_\mathcal{I}$.

Proof. We prove by induction on r that *(a)* implies *(c)* and that *(c)* implies *(b)*, the last implication being obvious. For $r = 1$, since no s-state is reachable, it amounts to the previous proof of NP-hardness.

Assume the result holds up to index r. Then, if $q_{r+1} \models \psi_{t+1}$ for some $t \geq r$, we pick a strategy for coalition AC witnessing this property. Again, we consider the different possible choices available to player D:

- if player D chooses to go to one of q_u and $\overline{q_u}$, with $u < r + 1$: the execution ends up in q_u if player C_u chose to set z_u to true. But in that case, formula ψ_{t+1} still holds in q_u, which yields by i.h. that z_u really evaluates to true in $v_\mathcal{I}$. Conversely, the execution ends up in $\overline{q_u}$ if player C_u set z_u to false. In that case, we get that $q_u \models \neg\psi_t$, with $t \geq u$, which entails by i.h. that z_u evaluates to false.
  This first case entails that player C_1 to C_r chose the correct value for variables z_1 to z_r.
- if player D chooses a set of eight states corresponding to a clause S^j_{r+1}, then the strategy of other players ensures that the execution will reach a state labeled with α. As in the previous reduction, this indicates that the corresponding clause has been made true by the choices of the other players.

Putting all together, this proves that variable z_{r+1} evaluates to true.

Now, if variable z_{r+1} evaluates to true, Assume the players in AC play according to valuation $f_{\mathcal{I}}$. Then

- if player D chooses to go to a set of states that correspond to a clause of φ_{r+1}, he will necessarily end up in a state that is labeled with α, since the clause is made true by the valuation we selected.
- if player D chooses to go to one of q_u or $\overline{q_u}$, for some u, then he will challenge player B_u to prove that his choice was correct. By i.h., and since player B_u played according to $f_{\mathcal{I}}$, formula $(\neg s)\,\mathbf{U}\,(\alpha \vee \mathbf{EX}\,(s \wedge \mathbf{EX}\,\neg\psi_{t+1}))$ will be satisfied, for any $t \geq u$. □

E Complexity of model checking ATL^+

Proposition 20. *Model checking ATL$^+$ can be achieved in Δ_3^{P} on implicit CGSs.*

Proof. A Δ_3^{P} algorithm is given in [18] for explicit CGSs. We extend it to handle implicit CGSs: for each subformula of the form $\langle\!\langle A\rangle\!\rangle\,\varphi$, guess (state-based) strategies for players in A. In each state, the choices of each player in A can be replaced in the transition functions. We then want to compute the set of states where the CTL^+ formula $\mathbf{A}\varphi$ holds. This can be achieved in Δ_2^{P} [8, 14], but requires to first compute the possible transitions in the remaining structure, *i.e.*, to check which of the transition formulae are satisfiable. This is done by a polynomial number of independent calls to an NP oracle, and thus does not increase the complexity of the algorithm. □

Proposition 21. *Model checking ATL$^+$ on turn-based two-player explicit CGSs is Δ_3^{P}-hard.*

Proof. This reduction is a quite straightforward extension of the one presented in [14] for CTL^+. In particular, it is quite different from the previous reductions, since the boolean formulae are now encoded in the ATL^+ formula, and not in the model.

We encode an instance $\mathcal{I}$ of SNSAT_2, keeping the notations used in the proofs of Prop. 12 (for the SNSAT_2 problem) and 16 (for clause numbering). Fig. 6 depicts the turn-based two-player CGS $\mathcal{C}$ associated to $\mathcal{I}$. States s_1 to s_m are labeled by atomic proposition s, states $\overline{z_1}$ to $\overline{z_m}$ are labeled by atomic proposition $\overline{z}$, and the other states are labeled by their name as shown on Fig. 6.

The ATL^+ formula is built recursively, with $\psi_0 = \top$ and

$$\psi_{k+1} = \langle\!\langle A\rangle\!\rangle\,[\mathbf{G}\,\neg s \wedge \mathbf{G}\,(\overline{z} \rightarrow \mathbf{EX}\,(s \wedge \mathbf{EX}\,\neg\psi_k)) \wedge \bigwedge_{w\leq p} [(\mathbf{F}\,z_w) \rightarrow \bigwedge_{j\leq n} \bigvee_{k\leq 3} \mathbf{F}\,l_w^{j,k}]]$$

where $l_w^{j,k} = v$ when $s_w^{j,k} = v$ and $\alpha_w^{j,k} = 1$, and $l_w^{j,k} = \overline{v}$ when $s_w^{j,k} = v$ and $\alpha_w^{j,k} = 0$. We then have:

Lemma 22. *For any $r \leq p$ and $t \geq r$, the following statements are equivalent:*

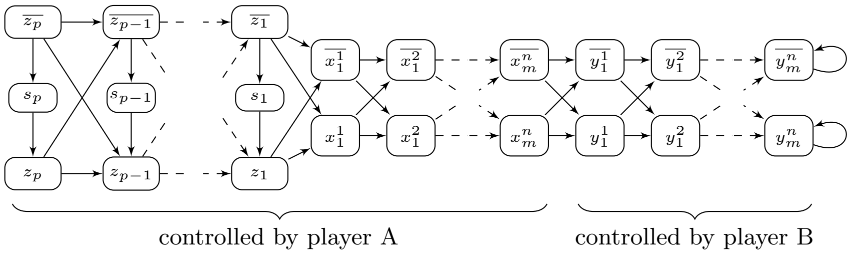


Fig. 6. The CGS $\mathcal{C}$

(a) $z_r \models \psi_t$*;*
(b) the strategies $f_{\mathcal{I}}$ *witness the fact that* $q_r \models \psi_t$*;*
(c) variable z_r *evaluates to true in* $v_{\mathcal{I}}$*.*

When $r = 1$, since no s- or $\overline{z}$-state is reachable from z_1, the fact that $z_1 \models \psi_t$, with $t \geq 1$, is equivalent to $z_1 \models \langle\!\langle A \rangle\!\rangle \bigwedge_j \bigvee_k \mathbf{F}\, l_1^{j,k}$. This in turn is equivalent to the fact that z_1 evaluates to true in $\mathcal{I}$.

We now turn to the inductive case. If $z_{r+1} \models \psi_{t+1}$ with $t \geq r$, consider a strategy for A s.t. all the outcomes satisfy the property, and pick one of those outcomes, say ρ. Since it cannot run into any s-state, it defines a valuation v_ρ for variables z_1 to z_{r+1} and x_1^1 to x_m^n in the obvious way. Each time the outcome runs in some $\overline{z_u}$-state, it satisfies $\mathbf{EX}\,(s \wedge \mathbf{EX}\,\psi_t)$. Each time it runs in some z_u-state, the suffix of the outcome witnesses formula ψ_{t+1} in z_u. Both cases entail, thanks to the i.h., that $v_\rho(z_u) = v_{\mathcal{I}}(z_u)$ for any $u < r + 1$. Now, the subformula $\bigwedge_w [(\mathbf{F}\, z_w) \rightarrow \bigwedge_{j \leq n} \bigvee_{k \leq 3} \mathbf{F}\, l_w^{j,k}$, when $w = r + 1$, entails that φ_{r+1} is indeed satisfied whatever the values of the y_{r+1}^j's, *i.e.*, that z_{r+1} evaluates to true in $\mathcal{I}$.

Conversely, if z_r evaluates to true, then strategy $f_{\mathcal{I}}$ clearly witnesses the fact that ψ_t holds in state z_r. □