ENS CACHAN

ATL with strategy contexts and bounded memory

Thomas Brihaye, Arnaud Da Costa,
François Laroussinie, Nicolas Markey

Research report LSV-08-14

Laboratoire Spécification et Vérification

CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE

École Normale Supérieure de Cachan
61, avenue du Président Wilson
94235 Cachan Cedex France

ATL with strategy contexts and bounded memory

Thomas Brihaye[1], Arnaud Da Costa[2],
François Laroussinie[3], and Nicolas Markey[2]

[1] Institut de mathématiques, Université de Mons-Hainaut, Belgium
[2] Lab. Spécification & Vérification, ENS Cachan – CNRS UMR 8643, France
[3] LIAFA, Univ. Paris 7 – CNRS UMR 7089, France
thomas.brihaye@umh.ac.be, dacosta@lsv.ens-cachan.fr,
francoisl@liafa.jussieu.fr, markey@lsv.ens-cachan.fr

Abstract. We extend the alternating-time temporal logics ATL and ATL* with *strategy contexts* and *memory constraints*: the first extension make strategy quantifiers to not "forget" the strategies being executed by the other players. The second extension allows strategy quantifiers to restrict to memoryless or bounded-memory strategies.
We first consider expressiveness issues. We show that our logics can express important properties such as equilibria, and we formally compare them with other similar formalisms (ATL, ATL* , Game Logic, Strategy Logic, ...). We then address the problem of model-checking for our logics, providing a PSPACE algoritm for the sublogics involving only memoryless strategies and an EXPSPACE algorithm for the bounded-memory case.

1 Introduction

Temporal logics and model checking. Temporal logics (LTL, CTL) have been proposed for the specification of reactive systems almost thirty years ago [15, 7, 16]. Since then, they have been widely studied and successfully used in many situations, especially for model checking—the automatic verification that a model of a system satisfies a temporal logic specification.

Alternating-time temporal logic. Over the last ten years, a new flavor of temporal logics has been developed: *alternating-time temporal logics* (ATL) [2]. ATL is a fundamental logic for specifying and verifying properties in *multi-agent systems* (modeled as *Concurrent Game Systems* (CGS) [2]), in which several agents can concurrently act upon the behaviour of the system. On these models, it is not only interesting to know if something *can* or *will* happen, as is expressed in CTL or LTL, but also if some agent(s) can *control* the evolution of the system in order to enforce a given property, whatever the other agents do. ATL can express this kind of properties thanks to its quantifier over strategies, denoted $\langle\!\langle A \rangle\!\rangle$ (where A is a coalition of agents). That coalition A has a strategy for reaching a winning location is then written $\langle\!\langle A \rangle\!\rangle \mathbf{F}\,\mathtt{win}$ (where $\mathbf{F}$ is the LTL modality for "eventually").

Our contributions. In this paper, we extend ATL and ATL* to express richer properties: while ATL strategy quantifier drops strategies introduced by earlier quantifiers in the evaluation of the formula, our logics keep executing those strategies. For example, our new modality, which we write $\langle\cdot A\rangle$, allows us to express the property "A has a strategy such that (1) Player B always has a strategy (given that of A) to enforce Φ and (2) Player C always has a strategy (given the *same* strategy of A) to enforce Ψ". This would be written as follows: $\langle\cdot A\rangle\,\mathbf{G}\,\big(\langle\cdot B\rangle\,\Phi\wedge\langle\cdot C\rangle\,\Psi\big)$. Note that naive attempts to express this property in ATL fail: for instance, in the ATL formula $\langle\!\langle A\rangle\!\rangle\,\mathbf{G}\,(\,\langle\!\langle B\rangle\!\rangle\,\Phi\wedge\langle\!\langle C\rangle\!\rangle\,\Psi)$, the coalitions do not cooperate anymore, while in $\langle\!\langle A\rangle\!\rangle\,\mathbf{G}\,(\,\langle\!\langle A,B\rangle\!\rangle\,\Phi\wedge\langle\!\langle A,C\rangle\!\rangle\,\Psi)$, the coalition A is allowed to use different strategies when playing with B and C. In order to achieve this idea, we naturally adapt the semantics of ATL* in order to interpret a formula within a stategy context.

Secondly we parameterise strategy quantifiers with the *resources* (in terms of memory) allowed for strategies: we define the quantifier $\langle\cdot A^s\cdot\rangle$ with $s\in(\mathbb{N}\cup\{\infty\})$, which restricts the quantification to strategies using memory of size s (called s-memory strategies) for Player A. It is well-known that memoryless strategies are enough to enforce ATL properties, but this is not the case for ATL* formulae, nor for our extension of ATL (and ATL*) with strategy contexts.

Our results are twofold: we mainly study the expressiveness of the new formalisms we propose, and compare them with classical formalisms such as ATL, ATL*, Alternating-time-μ-calculus (AMC) [2], the Game Logic (GL) [2], and also with recent formalisms such as Strategy Logic (SL) [6] or $\mathrm{Q}D_\mu$ [13], ... While our logic does not contain any of those two, it can express most of their interesting properties, especially regarding equilibria. We also study the model-checking problem for our logics: while we have a non-elementary algorithm for the general case, we propose a polynomial-space algorithm for model-checking our logic in the memoryless case, and extend it to an exponential-space algorithm for the bounded-memory setting.

Related works. Very recently, several works have focused on the same kind of extension of ATL, and come up with different solutions that we list below. Generally speaking, this leads to very expressive logics, able to express equilibrium properties. The counterpart is that those logics have very high complexity.

- IATL [1] extends ATL with strategy contexts, with a similar definition as ours, but it forces players to commit to a strategy, which they are not allowed to modify in the sequel. This logic is then studied in the memoryless case (which is proven to be a strict restriction to memory-based strategies).
- SL [6] extends temporal logics with first-order quantification over strategies. This extension has been defined and studied only in the two-player turn-based setting, where a non-elementary algorithm is proposed.
- $\mathrm{Q}D_\mu$ [13] considers strategies as labellings of the computation tree of the game structure with fresh atomic propositions. This provides a way of explicitly dealing with strategies. This extension is added on top of the decision μ-calculus $D\mu$, yielding a very expressive, yet decidable framework.

– Stochastic Game Logic [4] is a similar extension to ours, but for stochastic games. It is undecidable in the general case, but proved decidable when restricting to memoryless strategies.

Plan of the paper. Section 2 contains the definitions of our logics, and of our bounded-memory setting. Section 3 deals with the expressiveness results, and compares our extension with those cited in the related work above. In Section 4, we consider the model-checking problem for our extensions, and provide algorithms for the case of s-memory strategies.

Due to lack of space, some of the proofs are postponed to the appendix.

2 Definitions

In this section we introduce classical definitions of concurrent game structures, strategies and outcomes. We then define a notion of s-bounded memory strategies.

2.1 Concurrent Game Structures

Concurrent game structures are a multi-player extension of classical Kripke structures [2]. Their definition is as follows:

Definition 1. *A* Concurrent Game Structure *(*CGS *for short)* $\mathcal{C}$ *is a* 8*-tuple* $(\mathsf{Loc}, \ell_0, \mathsf{AP}, \mathsf{Lab}, \mathsf{Agt}, \mathcal{M}, \mathsf{Mov}, \mathsf{Edg})$ *where:*

- Loc *is a finite set of* locations*,* $\ell_0 \in \mathsf{Loc}$ *is the initial location;*
- AP *is a finite, non-empty set of atomic propositions;*
- $\mathsf{Lab}\colon \mathsf{Loc} \to 2^{\mathsf{AP}}$ *is a labelling function;*
- $\mathsf{Agt} = \{A_1, ..., A_k\}$ *is a finite set of* agents *(or* players*);*
- $\mathcal{M}$ *is a finite, non-empty set of moves;*
- $\mathsf{Mov}\colon \mathsf{Loc} \times \mathsf{Agt} \to \mathcal{P}(\mathcal{M}) \setminus \{\varnothing\}$ *defines the (finite) set of possible moves of each agent in each location.*
- $\mathsf{Edg}\colon \mathsf{Loc} \times \mathcal{M}^k \to \mathsf{Loc}$*, where* $k = |\mathsf{Agt}|$*, is a function defining the transition table. With each location and each set of moves of the agents, it associates the resulting location.*

The size $|\mathcal{C}|$ of a CGS $\mathcal{C}$ is defined as $|\mathsf{Loc}| + |\mathsf{Edg}|$, where $|\mathsf{Edg}|$ is the size of the transition table[1].

The intended behaviour is as follows [2]: in a location ℓ, each player A_i chooses one possible move m_{A_i} in $\mathsf{Mov}(\ell, A_i)$ and the next 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 $\mathsf{Next}(\ell, A_j, m)$, with $m \in \mathsf{Mov}(\ell, A_j)$, for the restriction of $\mathsf{Next}(\ell)$ to locations reachable from ℓ when player A_j makes the move m.

[1] Our results would still holds if we consider symbolic CGSs, where the transition table is encoded through boolean formulas [11].

2.2 Coalition, bounded-memory strategy, outcomes.

A coalition is a subset of agents. In multi-agent systems, a coalition A plays against its opponent coalition $\mathsf{Agt} \setminus A$ as if they were two single players. We thus extend Mov and Next to coalitions:

- Given $A \subseteq \mathsf{Agt}$ and $\ell \in \mathsf{Loc}$, $\mathsf{Mov}(\ell, A)$ denotes the possible moves for coalition A from ℓ. Such a move m is composed of a single move for every agent of the coalition, that is $m \stackrel{\text{def}}{=} (m_a)_{a \in A}$.
- Next is extended to coalitions in a natural way: given $m = (m_a)_{a \in A} \in \mathsf{Mov}(\ell, A)$, we let $\mathsf{Next}(\ell, A, m)$ denote the restriction of $\mathsf{Next}(\ell)$ to locations reachable from ℓ when every player $A_j \in A$ makes the move m_{A_j}.

Strategies and outcomes. Let $\mathcal{C}$ be a CGS. A *computation* of $\mathcal{C}$ is an infinite sequence $\rho = \ell_0 \ell_1 \ldots$ of locations such that for any i, $\ell_{i+1} \in \mathsf{Next}(\ell_i)$. We write ρ^i for the i-th suffix of ρ, and $\rho[i]$ for the $i+1$-st 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, *i.e.*, satisfying $f_{A_i}(\ell_0 \ldots \ell_m) \in \mathsf{Mov}(\ell_m, A_i)$. A strategy is *memoryless* if it only depends on the current state (*i.e.*, $f_{A_i}(\ell_0 \ldots \ell_m) = f_{A_i}(\ell_m)$). A strategy for a coalition A of agents is a set of strategies, one for each agent in the coalition. The set of strategies (resp. memoryless strategies) for A is denoted $\mathsf{Strat}(A)$ (resp. $\mathsf{Strat}^0(A)$).

A strategy for A_j induces a set of computations from ℓ, called the *outcomes* of f_{A_j} from ℓ and denoted $\mathsf{Out}(\ell, f_{A_j})$, that player A_j can enforce: $\ell_0 \ell_1 \ldots \in \mathsf{Out}(\ell, f_{A_j})$ iff $\ell_0 = \ell$ and $\ell_{i+1} \in \mathsf{Next}(\ell_i, A_j, f_{A_j}(\ell_0 \ldots \ell_i))$ for any i. Given a coalition A, a strategy for A is a tuple F_A containing one strategy for each player in A: $F_A = \{f_{A_j} | A_j \in A\}$. The *domain* of F_A ($\mathsf{dom}(F_A)$) is A. The strategy f_{A_j} for A_j is also denoted $(F_A)_{|A_j}$; more generally, $(F_A)_{|B}$ (resp. $(F_A)_{\setminus B}$) denotes the restriction of F_A to the coalition $A \cap B$ (resp. $A \setminus B$). The outcomes of F_A from a location ℓ are the computations enforced by the strategies in F_A: $\ell_0 \ell_1 \ldots \in \mathsf{Out}(\ell, F_A)$ iff $\ell_0 = \ell$ and for any i, $\ell_{i+1} \in \mathsf{Next}(\ell_i, A, (f_{A_j}(\ell_0, \ldots, \ell_i))_{A_j \in A})$. Note that $\mathsf{Out}(\ell, F_A) \subseteq \mathsf{Out}(\ell, (F_A)_{|B})$ for any coalitions A and B, and in particular that $\mathsf{Out}(\ell, F_\varnothing)$ represents the set of all computations from ℓ (since $\mathsf{Strat}(\varnothing) = \{\varnothing\}$).

It is also possible to *combine* two strategies $F \in \mathsf{Strat}(A)$ and $F' \in \mathsf{Strat}(B)$, resulting in a strategy $F \circ F' \in \mathsf{Strat}(A \cup B)$ defined as follows: $(F \circ F')_{|A_j}(\ell_0 \ldots \ell_m)$ equals $F_{|A_j}(\ell_0 \ldots \ell_m)$ if $A_j \in A$, and it equals $F'_{|A_j}(\ell_0 \ldots \ell_m)$ otherwise.

Finally, given a strategy F, an execution ρ and some integer $i \geq 0$, we define the restriction $F^{\rho,i}$ of F to the behavior from the future state $\rho[i]$: $F^{\rho,i}(\pi) = F(\rho[0 \ldots i] \cdot \pi)$. Note that if F is memoryless, then $F^{\rho,i} = F$.

Bounded-memory strategies. Between the general strategies (without any bound over its resources) and the simple memoryless strategies, we can consider *s-bounded memory strategies*. Let s be a (binary-encoded) integer representing the size of the memory. The most basic notion of memory-based strategies is based on the s-th suffix of the history: in that case, a strategy σ is a function from $Q^{[0,s]}$ to $\mathcal{M}$. The main drawback of this approach is that the players cannot choose

which part of the history is recorded, they can only remember the s last events (see Appendix A.1 for an example of this very limited kind of memory-bounded strategy).

More interestingly, we can define a bounded memory strategy as a memoryless strategy over the locations of the CGS **and** a set of memory cells [12]: choosing the move depends on both the location and the current memory cell, and after every move, the player can "update" its memory by moving to another cell. The size of the memory is then defined as the number of cells. Let Cell be the set of $s+1$ memory cells $\{0,\dots,s\}$.

Formally an s-memory strategy F_A for Player A is a 3-tuple $(F^{\mathsf{mov}}, F^{\mathsf{cell}}, c)$ where: F^{mov} is a mapping from $\mathsf{Cell}\times\mathsf{Loc}$ to $\mathcal{M}$ that associates a move with the current memory cell and the current location of the CGS, F^{cell} is a mapping from $\mathsf{Cell}\times\mathsf{Loc}$ to Cell that updates the memory cell, and c is the current memory cell of this strategy. In the following, we also denote $F^{\mathsf{mov}}(c,\ell)$ by $F_A(\ell)$ in order to simplify the notation: $F_A(\ell)$ is the move for A induced by the strategy F_A from the location ℓ and given then current memory cell c.

We can adapt the previous notions to this kind of strategy $F_A = (F^{\mathsf{mov}}, F^{\mathsf{cell}}, c)$. The set $\mathsf{Next}(\ell, A, F_A(\ell))$ contains the possible successor locations when A plays from ℓ according to F_A. Of course, the memory cell of F_A changes along an execution ρ, and we define $F_A^{\rho,i}$ as the strategy $(F^{\mathsf{mov}}, F^{\mathsf{cell}}, c_i)$ where c_i is defined inductively with: $c_0 = c$ and $c_{j+1} = F^{\mathsf{cell}}(\rho[j], c_j)$. Finally the outcomes $\mathsf{Out}(\ell, F_A)$ are the executions $\rho = \ell_0\ell_1\dots$ such that $\ell_{j+1} \in \mathsf{Next}(\ell_j, A, F_A^{\rho,j}(\ell_j))$.

Coalitions are handled the usual way: we use pairs $(A, \overline{s})$ to represent a coalition $A \subseteq \mathsf{Agt}$ and a memory-bounds vector $\overline{s} \in (\mathbb{N}\cup\{\infty\})^A$ which associates a size $\overline{s}(A_j)$ with the memory that agent $A_j \in A$ can use for its strategy. The set of strategies for A with memory bound $\overline{s}$ is denoted $\mathsf{Strat}^{\overline{s}}(A)$, and we omit to mention the memory bound when none is imposed.

2.3 The logic $\mathsf{ATL}^\star_{sc,\infty}$

We now define the logic $\mathsf{ATL}^\star_{sc,\infty}$ that extends $\mathsf{ATL}^\star$ with strategy contexts and bounded-memory strategy quantifiers:

Definition 2. *The syntax of $\mathsf{ATL}^\star_{sc,\infty}$ is defined by the following grammar:*

$$\mathsf{ATL}^\star_{sc,\infty} \ni \varphi_s, \psi_s ::= P \mid \neg\varphi_s \mid \varphi_s \vee \psi_s \mid \langle\!\langle A, \overline{s}\rangle\!\rangle\, \varphi_p \mid \rangle\!\rangle A \langle\!\langle\, \varphi_s$$
$$\varphi_p, \psi_p ::= \varphi_s \mid \neg\varphi_p \mid \varphi_p \vee \psi_p \mid \mathbf{X}\,\varphi_p \mid \varphi_p\,\mathbf{U}\,\psi_p$$

with $P \in \mathsf{AP}$, $A \subseteq \mathsf{Agt}$ and $\overline{s} \in (\mathbb{N}\cup\{\infty\})^A$. For convenience, we write e.g. $\langle\!\langle A^s, B^t\rangle\!\rangle\,\varphi$ instead of $\langle\!\langle \{A,B\}, \{A\mapsto s, B\mapsto t\}\rangle\!\rangle\,\varphi$.

Given a formula $\varphi \in \mathsf{ATL}^\star_{sc,\infty}$, the size of φ, denoted by $|\varphi|$, is the size of the tree representing that formula, assuming binary-encoding of integer constants.

We use standard abbreviations such as $\top \stackrel{\text{def}}{\equiv} P \vee \neg P$, $\bot \stackrel{\text{def}}{\equiv} \neg\top$, $\mathbf{F}\,\varphi \stackrel{\text{def}}{\equiv} \top\,\mathbf{U}\,\varphi$, etc, and omit to mention the memory bound when no memory constraint is imposed.

An $\mathsf{ATL}^\star_{sc,\infty}$ formula Φ is interpreted over a state ℓ of a CGS $\mathcal{C}$ *within a strategy context* $F \in \mathsf{Strat}(B)$ *for some coalition* B; this is denoted by $\ell \models_F \Phi$. The semantics is defined as follows:

$$
\begin{array}{rcl}
\ell \models_F \langle\!\cdot A, \overline{s} \cdot\rangle\!\rangle\, \varphi_p & \text{iff} & \exists F_A \in \mathsf{Strat}^{\overline{s}}(A).\ \forall \rho \in \mathsf{Out}(\ell, F_A \circ F).\ \rho \models_{F_A \circ F} \varphi_p, \\
\ell \models_F \cdot\rangle\!\rangle A \langle\!\langle \cdot\, \varphi_s & \text{iff} & \ell \models_{F_{\setminus A}} \varphi_s, \\
\rho \models_F \varphi_s & \text{iff} & \rho[0] \models_F \varphi_s, \\
\rho \models_F \mathbf{X}\, \varphi_p & \text{iff} & \rho^1 \models_{F^{\rho,1}} \varphi_p, \\
\rho \models_F \varphi_p\, \mathbf{U}\, \psi_p & \text{iff} & \exists i.\ \rho^i \models_{F^{\rho,i}} \psi_p \text{ and } \forall 0 \leq j < i.\ \rho^j \models_{F^{\rho,j}} \varphi_p.
\end{array}
$$

Given a CGS $\mathcal{C}$ with initial location ℓ_0, and an $\mathsf{ATL}^\star_{sc,\infty}$ formula Φ, the model-checking problem consists in deciding whether[2] $\ell_0 \models_\varnothing \Phi$.

$\mathsf{ATL}^\star_{sc,\infty}$ contains several extensions compared to $\mathsf{ATL}^\star$: the combination (or nesting) of strategies in a context and the way of handling separately $\overline{s}$-memory strategies.

Two $\mathsf{ATL}^\star_{sc,\infty}$ formulae Φ and Φ' are equivalent, denoted $\Phi \equiv \Phi'$, iff their truth value is the same for any location in any CGS under any strategy context F: $\ell \models_F \Phi \Leftrightarrow \ell \models_F \Phi'$ for any $\mathcal{C}$, ℓ, and F.

The formula $\langle\!\cdot A, \overline{s} \cdot\rangle\!\rangle\, \varphi$ holds on a location ℓ within a context F for a coalition B iff there exists a $\overline{s}$-memory strategy for A to enforce φ when B plays according to the strategy F. We use $\langle\!\langle \cdot A \cdot \rangle\!\rangle$ to denote the modality with no restriction over the memory allowed for the strategies of A (*i.e.*, the modality $\langle\!\langle \cdot A, \infty^A \cdot \rangle\!\rangle$); and we use $\langle\!\langle \cdot A^0 \cdot \rangle\!\rangle$ as an abbreviation for $\langle\!\langle \cdot A, 0^A \cdot \rangle\!\rangle$ to consider only memoryless strategies.

Conversely the modality $\cdot\rangle\!\rangle A \langle\!\langle \cdot$ removes the strategy for A from the current context under which the formula is interpreted. The operator $\cdot\rangle\!\rangle \mathsf{Agt} \langle\!\langle \cdot$ allows us to empty the current context, and then we clearly have: $\ell \models_F \cdot\rangle\!\rangle \mathsf{Agt} \langle\!\langle\, \varphi \Leftrightarrow \ell \models_{F'} \cdot\rangle\!\rangle \mathsf{Agt} \langle\!\langle\, \varphi$ for any context F and F'.

This entails that $\mathsf{ATL}^\star_{sc,\infty}$ contains $\mathsf{ATL}^\star$ (thus also $\mathsf{CTL}^\star$). Indeed the classical strategy quantifier of $\mathsf{ATL}^\star$, namely $\langle\!\langle A \rangle\!\rangle$, does not handle strategy context: $\langle\!\langle A \rangle\!\rangle\, \varphi$ holds for a location ℓ iff A has a strategy to enforce φ whatever the choices of $\mathsf{Agt} \setminus A$. Clearly $\langle\!\langle A \rangle\!\rangle\, \varphi$ is equivalent to $\cdot\rangle\!\rangle \mathsf{Agt} \langle\!\langle \cdot\, \langle\!\langle \cdot A \cdot \rangle\!\rangle\, \varphi$.

Clearly the existence of an $\overline{s}$-memory strategy for A to enforce φ entails the existence of an $\overline{s'}$-memory strategy if $\overline{s'} \geq \overline{s}$ (*i.e.*, $\overline{s'}(A_j) \geq \overline{s}(A_j)$ for all $A_j \in A$). Note that the converse is not true except for special cases such as ATL where memoryless strategies are sufficient (see [2, 17]).

Now we introduce several fragments of $\mathsf{ATL}^\star_{sc,\infty}$:

- $\mathsf{ATL}^\star_{sc,b}$ (with $b \in \mathbb{N}$) is the fragment of $\mathsf{ATL}^\star_{sc,\infty}$ where the quantifiers $\langle\!\cdot A, \overline{s} \cdot\rangle\!\rangle$ only use memory-bounds less or equal to b. In particular, $\mathsf{ATL}^\star_{sc,0}$ only allows memoryless strategies.
- $\mathsf{ATL}^\star_{sc}$ is the fragment of $\mathsf{ATL}^\star_{sc,\infty}$ where no restriction over the memory is allowed (any strategy quantifier deals with strategies with possibly infinite memory).

[2] The context can be omitted when it is empty, and we can directly write $\ell \models \Phi$.

– $\mathsf{ATL}_{sc,\infty}$ contains the formulae where every temporal modality is in the immediate scope of a strategy quantifier (*i.e.*, the path formulae are restricted to $\varphi_s \,\mathbf{U}\, \psi_s$, $\varphi_s \,\mathbf{R}\, \psi_s$ – R is the "Release" modality–, and $\mathbf{X}\, \varphi_s$). It follows from the above discussion that $\mathsf{ATL}_{sc,\infty}$ contains ATL and CTL. We also define the fragments $\mathsf{ATL}_{sc,b}$ and ATL_{sc} as above.

3 Expressiveness

In this section, we consider expressiveness issues to first illustrate the ability of $\mathsf{ATL}^{\star}_{sc,\infty}$ to state interesting properties and then to compare it with other classical formalisms.

3.1 Some interesting formulas of $\mathsf{ATL}^{\star}_{sc,\infty}$

First we introduce a new strategy quantifier in order to specify that "for any strategy of coalition A, *every* run in the corresponding outcome satisfies a formula φ":

$$[\![A]\!]\,\varphi \stackrel{\text{def}}{\equiv} \neg\,\langle\!\langle A\rangle\!\rangle\,\neg\,\langle\!\langle\varnothing\rangle\!\rangle\,\varphi.$$

It translates as follows:

$$\ell \models_F [\![A]\!]\,\varphi \quad \text{iff} \quad \forall F_A \in \mathsf{Strat}(A).\ \forall \rho \in \mathsf{Out}(\ell, F_A \circ F).\ \rho \models_{F_A \circ F} \varphi.$$

Note that this modality is not the dual of $\langle\!\langle A\rangle\!\rangle$ because we want an universal quantification over the runs in the outcomes. Thus it is different from the $[\![A]\!]$ in [2] which specifies the existence of *counter*-strategies for $\mathsf{Agt}\backslash A$ to ensure φ.

Examples of properties using strategy contexts. The new modalities $\langle\!\langle A\rangle\!\rangle$ allow us to express many interesting properties over the strategies of different players in a game. In particular, our logics can express the different examples that motivated the introduction of SL, $\mathrm{Q}D_\mu$ or IATL:

– We can express the fact that *"Player* 1 *can ensure* Φ_1 *while Player* 2 *plays to ensure* Φ_2*"* with the following formula:

$$\Psi = \langle\!\langle A_1\rangle\!\rangle\,[\![A_2]\!]\left(\left(\rangle A_1\langle\!\langle\ \langle\!\langle\varnothing\rangle\!\rangle\,\Phi_2\right) \Rightarrow \Phi_1\right). \tag{1}$$

When we interpret Ψ with the semantics of $\mathsf{ATL}^{\star}_{sc,\infty}$ we get:

$$\ell \models \Psi \quad \text{iff} \quad \exists F_1.\ \forall F_2.(\forall \rho_2 \in \mathsf{Out}(\ell, \{F_2\}).\ \rho_2 \models_{F_2} \Phi_2) \ \Rightarrow \\ (\forall \rho \in \mathsf{Out}(\ell, \{F_1, F_2\}).\ \rho \models_{\{F_1,F_2\}} \Phi_1) \tag{2}$$

The use of modality $\rangle A_1\langle\!\langle$ allows us to specify that Φ_1 has to be ensured by Player 1 only when Player 2 follows a *true* strategy to ensure (whatever Player 1 does) Φ_2.

– the *winning secure equilibrium* [5] is a special form of Nash equilibrium where both players can cooperate to satisfy an objective $\Phi_1 \wedge \Phi_2$, and if Player 1 (resp. 2) abandons the cooperation to ensure $\neg\Phi_2$ (resp. $\neg\Phi_1$) then Player 2 (resp. 1) can enforce $\neg\Phi_1$ (resp. $\neg\Phi_2$). The existence of such equilibrium can be stated as follows (Φ_1 and Φ_2 are assumed to be LTL formulas):

$$\langle\!\cdot A_1, A_2 \cdot\!\rangle \Big(\Phi_1 \wedge \Phi_2 \ \wedge \ [\!\cdot A_1 \cdot\!] \, (\neg\Phi_2 \Rightarrow \neg\Phi_1) \ \wedge \ [\!\cdot A_2 \cdot\!] \, (\neg\Phi_1 \Rightarrow \neg\Phi_2) \Big).$$

  Indeed this formula states that Players 1 and 2 can cooperate together to ensure the common objective, but there is no way for one player to enforce its own objective and the negation of the objective of the other player.
– Given two players A_1 and A_2 having their own objectives Φ_1 and Φ_2, the *Nash equilibrium* holds for two strategies F_1 and F_2 for players 1 and 2 respectively, if there is no "better" strategy F_1' for A_1 w.r.t. Φ_1 when Player 2 plays according to F_2, and vice versa for Player 2. This property characterises pairs of strategies. Given two strategies F_1 and F_2 in the strategy context F, the following formula holds for states ℓ within F iff F_1 and F_2 correspond to a Nash equilibrium in ℓ:

$$\Big((\, \langle\!\cdot A_1 \cdot\!\rangle \, \Phi_1) \Rightarrow \Phi_1 \ \wedge \ (\, \langle\!\cdot A_2 \cdot\!\rangle \, \Phi_2) \Rightarrow \Phi_2 \Big)$$

– The notion of *dominating* strategy can also be expressed: if F is a strategy for A_1, then a state ℓ satisfies the following formula within the strategy context F iff F is a dominating strategy from s w.r.t. the objective Φ: $(\, \langle\!\cdot \mathsf{Agt} \backslash A_1 \cdot\!\rangle \, \neg\Phi) \Rightarrow \neg \, \langle\!\cdot A_1 \cdot\!\rangle \, \Phi$. Indeed if the opponent can ensure $\neg\Phi$ when A_1 plays F, then it was not possible for A_1 to win.

Bounding the memory of the opponent. Our definition of bounded-memory strategies does not restrict the possible co-strategies for the opponents. However, it might sometimes be interesting to look for strategies that are winning only against *deterministic* bounded-memory opponents. This can be expressed in our logic by dualizing the bounded-memory quantifier: formula $\langle\!\cdot A \cdot\!\rangle \, [\!\cdot \mathsf{Agt} \setminus A, \overline{s} \cdot\!] \, \Phi))$ states that coalition A has a strategy enforcing Φ if the opponent coalition has $\overline{s}$-bounded memory.

Expressiveness of $\cdot\!\rangle A \langle\!\cdot$ quantifier. In the previous section, we illustrate the use of modality $\cdot\!\rangle A \langle\!\cdot$ by expressing the classical $\mathsf{ATL}^\star$ modality $\langle\!\langle A \rangle\!\rangle$ with $\cdot\!\rangle \mathsf{Agt} \langle\!\cdot \, \langle\!\cdot A \cdot\!\rangle$: we first forget the current strategy context and then quantify over the existence of a strategy for A: relaxing is necessary because it has to be a real strategy, *i.e.*, correct for any choice for the other agents. In fact, this modality does not add expressive power to $\mathsf{ATL}^\star_{sc,\infty}$:

Proposition 3. *For any $\mathsf{ATL}^\star_{sc,\infty}$ formula Φ, there exists a formula Ψ containing no $\cdot\!\rangle - \langle\!\cdot$ modality such that $\Phi \equiv \Psi$.*

Proof. Given a subset of agents $C \subseteq \mathsf{Agt}$ and $\Phi \in \mathsf{ATL}^{\star}_{sc,\infty}$, we define formula $\overline{\Phi}^{C}$ recursively as follows:

$$\overline{\langle\!\langle A\rangle\!\rangle \Phi}^{C} \stackrel{\text{def}}{=} \langle\!\langle A\rangle\!\rangle\, [\![C \setminus A]\!]\, \overline{\Phi}^{C\setminus A} \qquad \overline{[\![A]\!] \Phi}^{C} \stackrel{\text{def}}{=} \overline{\Phi}^{C\cup A}$$

$$\overline{\Phi \,\mathbf{U}\, \Psi}^{C} \stackrel{\text{def}}{=} \overline{\Phi}^{C} \,\mathbf{U}\, \overline{\Psi}^{C} \qquad \overline{\mathbf{X}\, \Phi}^{C} \stackrel{\text{def}}{=} \mathbf{X}\, \overline{\Phi}^{C}$$

$$\overline{\Phi \wedge \Psi}^{C} \stackrel{\text{def}}{=} \overline{\Phi}^{C} \wedge \overline{\Psi}^{C} \qquad \overline{\neg \Phi}^{C} \stackrel{\text{def}}{=} \neg \overline{\Phi}^{C}$$

$$\overline{P}^{C} \stackrel{\text{def}}{=} P$$

Now, for any strategy context F and any $C \subseteq \mathsf{dom}(F)$, and for any state q and any execution ρ, we have the following equivalences:

$$q \models_{F_{\setminus C}} \Phi \Leftrightarrow q \models_{F} \overline{\Phi}^{C} \qquad \rho \models_{F_{\setminus C}} \Phi_p \Leftrightarrow \rho \models_{F} \overline{\Phi_p}^{C}$$

The proof is done by structural induction over the formula (see Appendix A.2). Letting $\Psi = \overline{\Phi}^{\varnothing}$ yields the result. □

3.2 Comparison with other formalisms

Formal comparison of expressiveness. Two comparison criteria can be used to compare the expressive power of logics:

- The *distinguishing power* is the ability of a logic to distinguish models. Two models C_1 and C_2 are distinguished by L iff there exists a formula $\Phi \in L$ s.t. $C_1 \models \Phi$ and $C_2 \not\models \Phi$. We write $L_1 \leq_{dp} L_2$ when L_2 can distinguish any two models that can be distinguished by L_1.
- The *expressiveness* is the ability of a logic to express properties. We say that L_2 is more expressive than L_1 (written $L_1 \leq_{ex} L_2$) when for every L_1 formula, there exists an equivalent formula in L_2.

We define $<_{dp}$, $\equiv_{dp}$, $>_{ex}$ and $\equiv_{ex}$ in a standard way. Note that logics L_1 and L_2 can have the same distinguishing power (*i.e.*, $L_1 \leq_{dp} L_2$ and $L_2 \leq_{dp} L_1$) and differ on the expressiveness criterion. This happens *e.g.* for CTL and $\mathsf{CTL}^{\star}$.

Comparison with $\mathsf{ATL}^{\star}$. From the distinguishing power point of view, we have the following two results that emphasize the ability of our new modality $\langle\!\langle A\rangle\!\rangle$:

Proposition 4. *For any $b \in \mathbb{N}$, we have $\{\mathsf{ATL}_{sc,b},\ \mathsf{ATL}_{sc}\} >_{dp} \mathsf{ATL}^{\star}$.*

Proof. First note that $\mathsf{ATL}_{sc,0}$ and ATL_{sc} both contain ATL (remember that memoryless quantification is sufficient for ATL) and that ATL and $\mathsf{ATL}^{\star}$ have the same distinguishing power [3]. This entails $\{\mathsf{ATL}_{sc,0}, \mathsf{ATL}_{sc}\} \geq_{dp} \mathsf{ATL}^{\star}$.

The proof that $\mathsf{ATL}_{sc,0} >_{dp} \mathsf{ATL}^{\star}$ uses the fact formula $\langle\!\langle \mathsf{Agt}^{0}\rangle\!\rangle\, (\mathbf{X}\, \langle\!\langle \varnothing^{0}\rangle\!\rangle\, \mathbf{X}\, P)$ has no equivalent in $\mathsf{ATL}^{\star}$ (see Lemma 13 in Appendix A.3). This entails that $\mathsf{ATL}_{sc,b} >_{dp} \mathsf{ATL}^{\star}$ for any b. Finally, we have $\mathsf{ATL}_{sc} >_{dp} \mathsf{ATL}^{\star}$ because $\langle\!\langle A_1\rangle\!\rangle\, \mathbf{X}\, \big(\, \langle\!\langle A_2\rangle\!\rangle\, \mathbf{X}\, P \wedge \langle\!\langle A_2\rangle\!\rangle\, \mathbf{X}\, P'\big)$ cannot be expressed in $\mathsf{ATL}^{\star}$ (see Lemma 14 in Appendix A.3). □

As a direct corollary of the previous results and of the inclusion of ATL (resp. $\mathsf{ATL}^\star$) in $\mathsf{ATL}_{sc,\infty}$ (resp. $\mathsf{ATL}^\star_{sc,\infty}$), we obtain the following theorem:

Theorem 5. *For any $b \in \mathbb{N}$, we have:*

- $\{\mathit{ATL}_{sc,\infty}, \mathit{ATL}_{sc,b}, \mathit{ATL}_{sc}\} >_{ex} \mathit{ATL}$
- $\mathit{ATL}^\star_{sc,\infty} >_{ex} \mathit{ATL}^\star$
- $\mathit{ATL}^\star \not\geq_{ex} \{\mathit{ATL}_{sc,b}, \mathit{ATL}_{sc}\}$

Finally note also that we can the use of strategy contexts in ATL allows us to reach the expressive power of $\mathsf{CTL}^\star$:

Proposition 6. $\mathit{CTL}^\star \leq_{ex} \mathit{ATL}_{sc}$

The proof is given in Appendix A.3. Here we just illustrate the underlying translation with two examples:

- $\mathsf{E}\mathbf{G}\,(P \vee \mathbf{X}\,P)$ is equivalent to $\langle\!\langle \mathsf{Agt} \rangle\!\rangle\, \mathbf{G}\,(P \vee \langle\!\langle \varnothing \rangle\!\rangle\, \mathbf{X}\,P)$;
- $\mathsf{E}(\widetilde{\mathbf{F}}\,P \Rightarrow \widetilde{\mathbf{F}}\,P')$ is equivalent to $\langle\!\langle \mathsf{Agt} \rangle\!\rangle\, \mathbf{F}\,(\,\langle\!\langle \varnothing \rangle\!\rangle\, \mathbf{G}\,\neg P) \vee \langle\!\langle \mathsf{Agt} \rangle\!\rangle\, \mathbf{G}\,(\,\langle\!\langle \varnothing \rangle\!\rangle\, \mathbf{F}\,P')$.

Comparison with Game Logic. We now compare our new logics with *Game Logic*, proving that it is strictly less expressive than $\mathsf{ATL}^\star_{sc}$. *Game Logic* was introduced in [2] in order to express the module checking problem [10]. This logic is an extension of $\mathsf{ATL}^\star$ that allows us to deal explicitly with the execution tree induced by a strategy: given such a tree t, it is possible to quantify over the executions inside t and specify temporal properties. For example, the formula $\exists A.((\exists P\,\mathbf{U}\,P') \wedge (\forall \mathbf{F}\,P''))$ specifies that there exists a strategy F_A for A such that we have for the tree induced by F_A: (1) there exists a run verifying $P\,\mathbf{U}\,P'$ and (2) every run verifies $\mathbf{F}\,P''$. The definition of GL is given in Appendix A.5. We have the following result:

Theorem 7. $\mathit{ATL}_{sc} >_{ex} \mathit{GL}$

Proof (sketch). The proof is based on a translation from GL into ATL_{sc} (see Prop. 16 in appendix) and the fact that the ATL_{sc} formula $\langle\!\langle A_1 \rangle\!\rangle\, \mathbf{X}\,(\,\langle\!\langle A_2 \rangle\!\rangle\, \mathbf{X}\,b \wedge \langle\!\langle A_3 \rangle\!\rangle\, \mathbf{X}\,a)$ has no equivalent in GL (see Prop. 17 in appendix). □

Comparison with AMC. Alternating-time μ-calculus is neither more nor less expressive than our extensions of ATL:

Proposition 8. $\mathit{ATL}_{sc,\infty} \not\leq_{ex} \mathit{AMC}$ *and* $\mathit{AMC} \not\leq_{ex} \mathit{ATL}^\star_{sc,\infty}$.

Proof. The CGS $\mathcal{C}_2$ (described in Figure 3 in Appendix A.3) is a simple unfolding of $\mathcal{C}_1$, thus they satisfy the same AMC formulae but they can be distinguished by $\mathsf{ATL}_{sc,\infty}$, this gives the first result. When considering one-player CGS (*i.e.*, Kripke structures), our $\mathsf{ATL}^\star_{sc,\infty}$ is clearly equivalent to $\mathsf{CTL}^\star$, which is strictly less expressive than the (classical) μ-calculus. □

Comparison with Strategy Logic [6]. Strategy Logic (SL for short) has recently been defined in [6] as an extension of LTL with first-order quantification on strategies. That player A has a strategy to enforce φ is then written $\exists \sigma_A.\ \forall \sigma_B.\ \varphi(\sigma_A, \sigma_B)$ where the arguments given to φ indicate on which path φ is evaluated.

While this logic has only been defined on 2-player turn-based games, its definition can easily be extended to our n-player CGS framework. We conjecture that $\mathsf{ATL}_{sc,\infty}$ and SL are incomparable:

- SL can explicitly manipulate strategies as first-order elements. It can for instance state properties such as

$$\exists x_1.\ \exists y_1.\ \exists x_2.\ \exists y_2.\ \big[\varphi_1(x_1, y_1) \wedge \varphi_2(x_2, y_1) \wedge \varphi_3(x_1, y_2) \wedge \varphi_4(x_2, y_2)\big]$$

  which (we conjecture) $\mathsf{ATL}_{sc,\infty}$ cannot express due to the circular constraint.
- on the other hand, SL requires subformulas embedded in modalities to be closed. As a consequence, formula

$$\exists x_1.\ \forall y_1.\ [\mathbf{G}\,(\exists y_2.\ [\mathbf{F}\,p](x_1, y_2))](x_1, y_1)$$

  is not an SL formula (because $\exists y_2.\ [\mathbf{F}\,p](x_1, y_2)$ is not closed), while it is expressed in $\mathsf{ATL}_{sc,\infty}$ as

$$\langle\!\cdot A \cdot\!\rangle\, \mathbf{G}\, (\, \langle\!\cdot B \cdot\!\rangle\, \mathbf{F}\, p).$$

However, it should be noticed that the simple one-alternation fragment of SL can be translated into $\mathsf{ATL}^{\star}_{sc,\infty}$.

Comparison with qD_μ [13]. The logic $\mathrm{Q}D_\mu$ [13] extends the decision μ-calculus with quantification over strategies. In that setting, strategies are thought of as a labelling of the infinite computation tree of the system with extra atomic propositions and this allows to have strategy contexts. There is neither explicit modality $\rangle\!\cdot A \cdot\!\langle$ nor bounded-memory strategy quantification in $\mathrm{Q}D_\mu$. But as we have seen above, any $\mathsf{ATL}^{\star}_{sc}$ formula can be translated into an equivalent $\cdot\rangle\!-\!\langle\cdot$-free formula, which can in turn be expressed in $\mathrm{Q}D_\mu$ [14].

Note also that $\mathrm{Q}D_\mu$ is able to express that a strategy is memoryless (provided that it has access to the names of the locations of the CGS): this is achieved by saying that the strategy always labels the same set of successors in any two copies of the same location in the execution tree.

Comparison with IATL [1]. IATL extends ATL with a context in a similar way as $\mathsf{ATL}_{sc,\infty}$ does, but there is no way of replacing a strategy by another one: once a player has committed to applying a strategy, she cannot change her mind. It is thus easy to see that $\mathsf{ATL}_{sc,\infty}$ is (strictly) more expressive than IATL.

Figure 1 summarizes the expressiveness results of our logics.

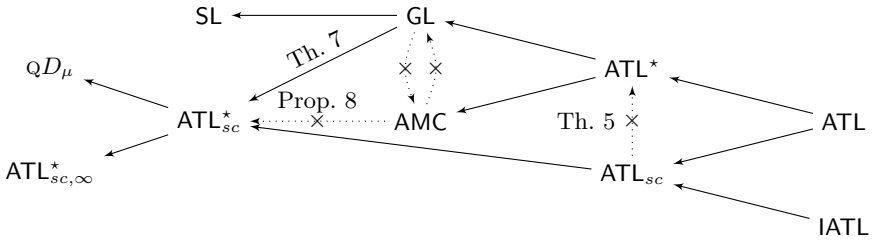

Fig. 1. Expressiveness of $\mathsf{ATL}_{sc,\infty}$ and $\mathsf{ATL}^\star_{sc,\infty}$ compared to classical logics

4 $\mathsf{ATL}_{sc,\infty}$ and $\mathsf{ATL}^\star_{sc,\infty}$ model-checking

We begin with proving that model-checking is decidable for our logic. Still, as is the case for Strategy Logic, the resulting algorithm has very high complexity. We thus focus on simpler cases (namely, memoryless and bounded-memory strategies), where more efficient algorithms can be obtained.

Theorem 9. *Model checking $\mathsf{ATL}^\star_{sc,\infty}$ formulas over CGS is decidable.*

Proof. The translation from $\mathsf{ATL}^\star_{sc}$ to $\mathrm{Q}D_\mu$ yields decidability of $\mathsf{ATL}^\star_{sc}$. Moreover, as we will see in Section 4.2, it is possible to encode the bounded-memory strategies as memoryless strategies over an extended CGS. Since memorylessness can be expressed with $\mathrm{Q}D_\mu$, this provides an indirect algorithm for $\mathsf{ATL}^\star_{sc,\infty}$ model checking. □

4.1 Model-checking $\mathsf{ATL}^\star_{sc,0}$ and $\mathsf{ATL}_{sc,0}$

Theorem 10. *The model checking problems for $\mathsf{ATL}^\star_{sc,0}$ and $\mathsf{ATL}_{sc,0}$ over CGSs are* PSPACE*-complete.*

Proof. We only address the membership in PSPACE. The hardness proof is given in Lemma 18 in Appendix A.6 (and is similar to that of [4]).

Let $\mathcal{C}$ be a CGS, ℓ a location and F a memoryless strategy context, assigning a memoryless strategy to each player of some coalition A. Since F contains only memoryless strategies, it associates with each location one move for each agent in A. Dropping the other moves of those agents, we get a CGS, denoted $(\mathcal{C}, F)$, whose set of executions is exactly the set of outcomes of F in $\mathcal{C}$.

From this and the fact that a memoryless strategy can be stored in space $O(|Q|)$, we get a simple PSPACE model-checking algorithm for $\mathsf{ATL}^\star_{sc,0}$ that relies on a (PSPACE) model-checking algorithm for LTL. The main difficulty is that strategy contexts prevent us from proceeding in a standard bottom-up fashion. As a consequence, our algorithm consists in enumerating strategies starting from outermost strategy quantifiers.

If φ is an $\mathsf{ATL}^\star_{sc,0}$ path formula, we denote by $\Phi(\varphi)$ the set of outermost *quantified* φ subformulae (*i.e.* of the form $\langle\!\langle A \rangle\!\rangle \psi$), and by $\sigma(\varphi)$ the corresponding

LTL formula where all subformulae $\psi \in \Phi(\varphi)$ have been replaced by new propositions a_ψ. We enumerate all possible contexts, recursively calling the algorithm at each step of the enumeration, and thus gradually taking care of each labelling a_ψ. Algorithm 1 describes the procedure. □

Algorithm 1 : MC-ATL$^\star_{sc,0}(\mathcal{C}, F, \ell_0, \varphi)$ – ATL$^\star_{sc,0}$ model checking

Require: a *CGS* $\mathcal{C}$, $F \in \mathsf{Strat}^0(A)$, $l_0 \in \mathsf{Loc}$ and an ATL$^\star_{sc,0}$ path formula φ
Ensure: YES iff $\forall \lambda \in \mathsf{Out}(\ell_0, F), \lambda \models_F \varphi$
  $\mathcal{C}' := (\mathcal{C}, F)$
  foreach $\psi \in \Phi(\varphi)$ **do**
    case $\psi = \langle\!\cdot B^0 \cdot\rangle\, \psi'$:
      for $F_B \in \mathsf{Strat}^0(B)$, $\ell \in \mathsf{Loc}$ **do**
        if MC-ATL$^\star_{sc,0}(\mathcal{C}, F_B \circ F, l, \psi')$, **then** label l with a_ψ
    case $\psi = \cdot B \langle\!\cdot\, \psi'$:
      for $l \in \mathsf{Loc}$ **do**
        if MC-ATL$^\star_{sc,0}(\mathcal{C}, F_{\setminus B}, l, \psi')$, **then** label l with a_ψ
  return MC LTL $(\mathcal{C}', l_0, \mathbf{A}\sigma(\varphi))$

Note that PSPACE-completeness extends straightforwardly to "memoryless" extensions (*i.e.*, with quantification over memoryless strategies) of ATL* and SL. Since ATL objectives do not require memory, ATL$_0$ is the same as ATL, and its model-checking problem is PTIME-complete. Moreover a similar algorithm would work for *symbolic CGSs*, a succinct encoding of CGS proposed in [9].

Remark 1. Both the algorithm and the PSPACE-hardness can be adapted for IATL. This corrects the Δ_2^{P}-completeness result of [1].

4.2 Bounded-memory strategies

The case of bounded-memory strategies can be handled in a similar way as memoryless strategies. Indeed as explained in Subsection 2.2, we can see an s-bounded strategy for Player A_i as a memoryless strategy over an extended structure containing the original CGS $\mathcal{C}$ and a particular CGS controlled by A_i and describing its memory.

Formally, for a player A_i, we define the CGS $\mathbb{M}^s_{A_i}$ as follows: $\mathbb{M}^s_{A_i} = (\mathsf{Agt}, \mathsf{Loc}^i_s, 0, \varnothing, \varnothing, \mathcal{M}^i_s \cup \{\bot\}, \mathsf{Mov}^i_s, \mathsf{Edg}^i_s)$ where

- $\mathsf{Loc}^i_s = \{0, \ldots, s\}$ is the set of (unlabeled) locations;
- $\mathcal{M}^i_s$ is isomorphic to Loc^i_s (and we identify both sets),
- Mov^i_s and Edg^i_s do not depend on the location: Mov^i_s allows only one move $\bot$ to each player, except for player A_i, who is allowed to play any move in $\mathcal{M}^i_s$. Then Edg^i_s returns the location chosen by A_i.

Let $\overline{s} \in \mathbb{N}^{\mathsf{Agt}}$ be a memory-bounds vector. Now considering the product structure $\mathcal{C}_{\overline{s}} = \prod_{A_i \in \mathsf{Agt}} \mathbb{M}_{A_i}^{\overline{s}(A_i)} \times \mathcal{C}$, for all player A_j we can very simply export $\overline{s}(A_j)$ memory-bounded strategies of $\mathcal{C}$ to *some* memoryless strategies over $\mathcal{C}_{\overline{s}}$. Indeed, given a Player A_j, we do not want to consider all memoryless strategies f over $\mathcal{C}_{\overline{s}}$ but only the ones where A_j exclusively uses the information from $\mathbb{M}_{A_j}^{\overline{s}(A_j)}$ (i.e., such that $f(i_1, \ldots, i_j, \ldots, i_k, l) = f(0, \ldots, i_j, \ldots, 0, l)$). Let $\mathsf{RStrat}^0_{\mathcal{C}_{\overline{s}}}(A_j)$ be this restricted set of such strategies ; clearly we have $\mathsf{RStrat}^0_{\mathcal{C}_{\overline{s}}}(A_j) \subset \mathsf{Strat}^0_{\mathcal{C}_{\overline{s}}}(A_j)$.

Adapting the proof of Theorem 10 to memory-bounded strategies, we get:

Proposition 11. *Let $C = (\mathsf{Agt}, \mathsf{Loc}, \ell_0, \mathsf{AP}, \mathsf{Lab}, \mathcal{M}, \mathsf{Mov}, \mathsf{Edg})$ be a CGS. Let $\varphi \in \mathsf{ATL}^\star_{sc,b}$ involving only $\overline{s}$-memory quantifiers. Then φ can be checked in exponential space.*

Proof. We run the algorithm of Theorem 10 over the structure $\mathcal{C}_{\overline{s}}$, restricting the enumerations of $\mathsf{Strat}^0_{\mathcal{C}_{\overline{s}}}(B)$ to those of $\mathsf{RStrat}^0_{\mathcal{C}_{\overline{s}}}(B)$. □

Remark 2.
- If the memory-bounds $\overline{s}$ were given in unary, our algorithm would be PSPACE, since the LTL model-checking over the product structure can be performed on the fly.
- Note that this algorithm can deal with formula containing several subformulas $\langle\!\langle A, \overline{s_1} \rangle\!\rangle \varphi_1, \ldots, \langle\!\langle A, \overline{s_p} \rangle\!\rangle \varphi_p$ with different different memory-bounds s_i (for the same coalition A).
- Since our algorithm consists in enumerating the strategies, it could cope with games of incomplete information, where the strategies would be based on (some of) the atomic propositions labeling a location, rather than on the location itself [17].
- Bounded-memory quantification can be defined also for the other formalisms where memory-based strategies are needed, *e.g.* $\mathsf{ATL}^\star$ or SL. Our EXPSPACE algorithm could easily be adapted to that case.

5 Conclusion

In this paper we propose powerful extensions of ATL and $\mathsf{ATL}^\star$ logics. These extensions allow us to express many interesting and complex properties that have motivated the definition of new formalisms in the past. An advantage of these extensions is to treat strategies through modalities as in ATL and $\mathsf{ATL}^\star$.

As future work, we plan to study the exact complexity of model-checking $\mathsf{ATL}_{sc,\infty}$ and $\mathsf{ATL}^\star_{sc,\infty}$, with the aim of possibly finding reasonably efficient algorithms for our expressive extensions of ATL and $\mathsf{ATL}^\star$. Finally we think that the ability to deal explicitly with bounded-memory strategies is an interesting approach to develop.

References

1. T. Ågotnes, V. Goranko, and W. Jamroga. Alternating-time temporal logics with irrevocable strategies. In *Proceedings of the 11th Conference on Theoretical Aspects of Rationality and Knowledge (TARK'07)*, pages 15–24, 2007.

2. R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49(5):672–713, 2002.
3. R. Alur, T. A. Henzinger, O. Kupferman, and M. Y. Vardi. Alternating refinement relations. In *Proc. 9th Intl Conf. Concurrency Theory (CONCUR'98)*, volume 1466 of *LNCS*, pages 163–178. Springer, 1998.
4. Ch. Baier, T. Brázdil, M. Größer, and A. Kučera. Stochastic game logic. In *Proceedings of the 4th International Conference on Quantitative Evaluation of Systems (QEST'07)*, pages 227–236. IEEE Comp. Soc. Press, 2007.
5. K. Chatterjee, T. A. Henzinger, and M. Jurdzinski. Games with secure equilibria. *Theoretical Computer Science*, 365(1-2):67–82, 2006.
6. K. Chatterjee, T. A. Henzinger, and N. Piterman. Strategy logic. In *Proc. of the 18th International Conference on Concurrency Theory (CONCUR'07), Lisbon, Portugal*, volume 47013 of *LNCS*, pages 59–73. Springer, 2007.
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)*, volume 131 of *LNCS*, pages 52–71. Springer, 1981.
8. E. A. Emerson. Temporal and modal logic. In *Handbook of Theoretical Computer Science*, volume B, chapter 16, pages 995–1072. Elsevier, 1990.
9. W. Jamroga and J. Dix. Do agents make model checking explode (computationally)? In *Proceedings of the 4th International Central and Eastern European Conference on Multi-Agent Systems (CEEMAS'05)*, volume 3690 of *LNCS*. Springer, 2005.
10. O. Kupferman, M. Y. Vardi, and P. Wolper. Module checking. *Information and Computation*, 164(2):322–344, 2001.
11. F. Laroussinie, N. Markey, and G. Oreiby. On the expressiveness and complexity of ATL. In *Proc. 10th Intl Conf. Foundations of Software Science and Computation Structures (FoSSaCS'07)*, volume 4423 of *LNCS*, pages 243–257, Braga, Portugal, 2007. Springer.
12. R. Mazala. Infinite games. In *Automata, Logics, and Infinite Games*, volume 2500 of *LNCS*, pages 23–42. Springer-Verlag, 2002.
13. S. Pinchinat. A generic constructive solution for concurrent games with expressive constraints on strategies. In *Proceedings of the 5th International Symposium on Automated Technology for Verification and Analysis (ATVA'07)*, LNCS. Springer-Verlag, 2007.
14. S. Pinchinat. Personal communication, 2008.
15. A. Pnueli. The temporal logic of programs. In *Proc. 18th Ann. Symp. Foundations of Computer Science (FOCS'77)*, pages 46–57. IEEE Comp. Soc. Press, 1977.
16. J.-P. Queille and J. Sifakis. Specification and verification of concurrent systems in CESAR. In *Proc. 5th Intl Symp. on Programming (SOP'82)*, volume 137 of *LNCS*, pages 337–351. Springer, 1982.
17. P.-Y. Schobbens. Alternating-time logic with imperfect recall. In *Proceedings of the 1st Workshop on Logic and Communication in Multi-Agent Systems (LCMAS'03)*, volume 85 of *ENTCS*. Elsevier, 2004.

A Appendix

A.1 Discussion about the bounded-memory strategy

Here we illustrate the drawback of the basic notion of bounded-memory strategy where we can only consider the s-th suffix of the history.

Consider the game depicted on Fig. 2. It can be proved that there exists no history-based bounded-memory strategy for Player 1 from state s_0 for the LTL objective $(\mathbf{X}\, a \wedge \mathbf{G}\, b) \vee (\mathbf{X}\, \neg a \wedge \mathbf{G}\, c)$ (stating that only b-states should be visited if a has been visited at the beginning of the execution, and only c-states should be visited otherwise) because the crucial information (visiting s_1) occurs at the very beginning of the execution.

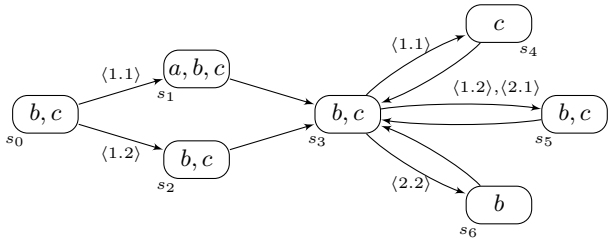

Fig. 2. A game with no history-based bounded-memory strategy

But with the notion of bounded-memory strategy we have adopted, we can easily see that Player 1 has a 1-bounded memory strategy to ensure that the formula $(\mathbf{X}\, a \wedge \mathbf{G}\, b) \vee (\mathbf{X}\, \neg a \wedge \mathbf{G}\, c)$ holds. Indeed, we can define the strategy $F = (F^{\mathsf{mov}}, F^{\mathsf{cell}}, 0)$ with:

$$F^{\mathsf{cell}}(0, s_0) = F^{\mathsf{cell}}(0, s_2) = F^{\mathsf{cell}}(0, s_3) = F^{\mathsf{cell}}(0, s_4) = F^{\mathsf{cell}}(0, s_5) = 0$$
$$F^{\mathsf{cell}}(0, s_1) = F^{\mathsf{cell}}(1, s_3) = F^{\mathsf{cell}}(1, s_5) = F^{\mathsf{cell}}(1, s_6) = 1$$
$$F^{\mathsf{mov}}(0, s_3) = 2 \text{ and } F^{\mathsf{mov}}(1, s_3) = 1$$

Memory cell 0 encodes the fact that s_1 has not (or "not yet") been visited. This strategy is easily seen to be winning.

A.2 Elimination of $\rangle A \langle$

Here we give the proof of the following lemma used in Proposition 3:

Lemma 12. *For any strategy context F, any subset $C \subseteq \mathsf{dom}(F)$ and any formula $\Phi \in \mathsf{ATL}^{\star}_{sc,\infty}$ and any path formula Φ_p, we have:*

$$q \models_{F_{\setminus C}} \Phi \quad \Leftrightarrow \quad q \models_F \overline{\Phi}^C$$
$$\rho \models_{F_{\setminus C}} \Phi_p \quad \Leftrightarrow \quad \rho \models_F \overline{\Phi_p}^C$$

Proof. The proof is done by structural induction over the formula. In this proof we will use C' as an abbreviation for the coalition $C \setminus A$. Moreover F_B ranges over strategies for coalition B.

– $\Psi \stackrel{\text{def}}{=} \langle\!\cdot A \cdot\!\rangle\, \varphi$.

We have the following equivalences : $q \models_F \langle\!\cdot A \cdot\!\rangle\, [\![C']\!]\, \overline{\varphi}^{C'}$ means by definition

$$\exists F_A.\ \forall F_{C'}.\ \forall \rho \in \mathsf{Out}(q, F_{C'} \circ F_A \circ F).\ \rho \models_{F_{C'} \circ F_A \circ F} \overline{\varphi}^{C'},$$

Then the induction hypothesis yields

$$\exists F_A.\ \forall F_{C'}.\ \forall \rho \in \mathsf{Out}(q, F_{C'} \circ F_A \circ F).\ \rho \models_{(F_{C'} \circ F_A \circ F)_{\setminus (C')}} \varphi,$$

or equivalently, since $(F_{C'} \circ F_A \circ F)_{\setminus C'} = F_A \circ (F_{\setminus C})$:

$$\exists F_A.\ \forall F_{C'}.\ \forall \rho \in \mathsf{Out}(q, F_{C'} \circ F_A \circ F).\ \rho \models_{F_A \circ (F_{\setminus C})} \varphi,$$

or else, since we have

$$\bigcup_{F_{C'} \in \mathsf{Strat}(C')} \mathsf{Out}(q, F_{C'} \circ F_A \circ F) = \mathsf{Out}(q, F_A \circ (F_{\setminus C}))$$

$$\exists F_A.\ \forall \rho \in \mathsf{Out}(q, F_A \circ (F_{\setminus C})).\ \rho \models_{(F_A \circ (F_{\setminus C}))} \varphi,$$

leading to $q \models_{F_{\setminus C}} \langle\!\cdot A \cdot\!\rangle\, \varphi$, which is the desired result.

– $\Psi \stackrel{\text{def}}{=} \rangle\!\cdot A \cdot\!\langle\, \varphi$. On the one hand, by the semantics of $\mathsf{ATL}^{\star}_{sc,\infty}$, we have that:

$$q \models_{F_{\setminus C}} \rangle\!\cdot A \cdot\!\langle\, \varphi \quad \text{iff} \quad q \models_{F_{\setminus (C \cup A)}} \varphi.$$

On the other hand, the induction hypothesis tells us that:

$$q \models_{F_{\setminus (C \cup A)}} \varphi \quad \text{iff} \quad q \models_F \overline{\varphi}^{C \cup A}.$$

Gathering the two equivalences, we obtain the desired result.

– $\Psi \stackrel{\text{def}}{=} \varphi\, \mathbf{U}\, \psi$. The semantics of $\mathsf{ATL}^{\star}_{sc,\infty}$ tells us that $\rho \models_{F_{\setminus C}} \Psi$ if and only if the following formal holds:

$$\exists i.\ \rho^i \models_{(F_{\setminus C})^{\rho,i}} \psi \text{ and } \forall 0 \leq j < i.\ \rho^j \models_{(F_{\setminus C})^{\rho,j}} \varphi.$$

By using the induction hypothesis, the above formula is equivalent to the following one:

$$\exists i.\ \rho^i \models_{F^{\rho,i}} \overline{\psi}^{C} \text{ and } \forall 0 \leq j < i.\ \rho^j \models_{F^{\rho,j}} \overline{\varphi}^{C},$$

which means that $\rho \models_F \overline{\varphi}^{C}\, \mathbf{U}\, \overline{\psi}^{C}$. We thus obtain the desired result.

– The remaining cases are straightforward.

□

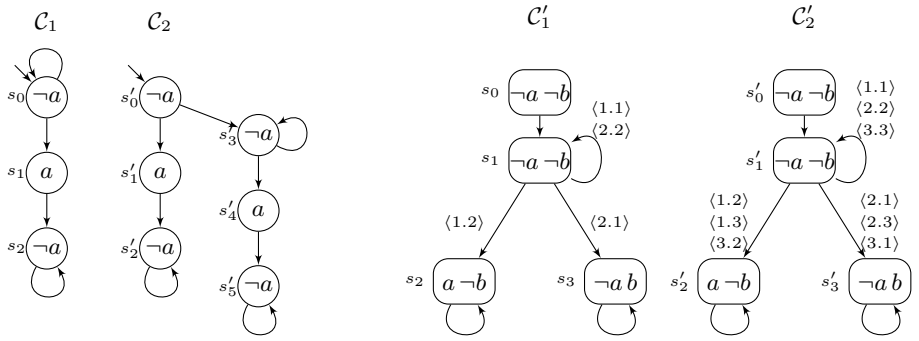

Fig. 3. $\mathcal{C}_1$ and $\mathcal{C}_2$ (resp $\mathcal{C}'_1$ and $\mathcal{C}'_2$) cannot be distinguished by ATL^*

A.3 Distinguishing power

Lemma 13. *$\langle\!\langle Agt^0\rangle\!\rangle\,(\mathbf{X}\,\langle\!\langle\varnothing^0\rangle\!\rangle\,\mathbf{X}\,P)$ has no equivalent in $\mathsf{ATL}^\star$.*

Proof. Consider the two one-player CGS $\mathcal{C}_1$ and $\mathcal{C}_2$ in Fig. 3. They clearly satisfy the same $\mathsf{ATL}^\star$ formulae since $\mathcal{C}_2$ corresponds to an unfolding of C_1. But we have:

$$s_0 \not\models \langle\!\langle \mathsf{Agt}^0\rangle\!\rangle\,\mathbf{X}\,(\,\langle\!\langle\varnothing^0\rangle\!\rangle\,\mathbf{X}\,a) \qquad s'_0 \models \langle\!\langle \mathsf{Agt}^0\rangle\!\rangle\,\mathbf{X}\,(\,\langle\!\langle\varnothing^0\rangle\!\rangle\,\mathbf{X}\,a)$$

Indeed any memoryless strategy in s_0 will contain either the infinite path $s_0 \to s_0 \to s_0 \to \ldots$, or the path $s_0 \to s_1 \to s_2 \to s_2 \to \ldots$ In order to have a after two transitions, it is necessary to first take the transition $s_0 \to s_0$ and secondly the transition $s_0 \to s_1$. This can only be achieved by a strategy with memory. On the contrary, s'_0 satisfies clearly the property. □

Note that we could also consider the logic $\mathsf{ATL}^\star_0$, that is the logic $\mathsf{ATL}^\star$ where every strategy quantifier deals with memoryless strategies. This logic could distinguish the two structures of the previous proof with the formula $\langle\!\langle \mathsf{Agt}\rangle\!\rangle_0\,\mathbf{X}\,\mathbf{X}\,P$.

Thus $\mathsf{ATL}_{sc,\infty}$ allows us to distinguish models that cannot be distinguished by $\mathsf{ATL}^\star$, and note that it is not necessary to use the modality $\rangle - \langle$ to obtain this result. As a consequence we have:

Lemma 14. *$\langle\!\langle A_1\rangle\!\rangle\,\mathbf{X}\left(\,\langle\!\langle A_2\rangle\!\rangle\,\mathbf{X}\,P \wedge \langle\!\langle A_2\rangle\!\rangle\,\mathbf{X}\,P'\right)$ has no equivalent in $\mathsf{ATL}^\star$.*

Proof. Now consider the CGSs $\mathcal{C}'_1$ and $\mathcal{C}'_2$ in Fig. 3. They satisfy the same $\mathsf{ATL}^\star$ formulae: in $\mathcal{C}'_2$, a third move is possible for both players but it does not give more strategies to enforce particular $\mathsf{ATL}^\star$ properties. But we clearly have that s'_0 satisfies $\langle\!\langle A_1\rangle\!\rangle\,\mathbf{X}\left(\,\langle\!\langle A_2\rangle\!\rangle\,\mathbf{X}\,P \wedge \langle\!\langle A_2\rangle\!\rangle\,\mathbf{X}\,P'\right)$ thanks to the third move of Player 1. But this is not the case for s_0 that does not satisfy the $\mathsf{ATL}_{sc,\infty}$ formula. □

A.4 Translation from $\mathsf{CTL}^\star$ to ATL_{sc}

Here we give the proof of Proposition 6:

Proof. Let Ψ be a $\mathsf{CTL}^\star$ formula. We first define a translation to build an "almost ATL_{sc}" formula $\overline{\Psi}$ as follows:

$$\overline{P} \stackrel{\text{def}}{=} P \qquad \overline{\mathbf{X}\,\varphi} \stackrel{\text{def}}{=} \langle\cdot\varnothing\cdot\rangle\,\mathbf{X}\,\overline{\varphi}$$
$$\overline{\neg\varphi} \stackrel{\text{def}}{=} \overline{\varphi} \qquad \overline{\varphi\,\mathbf{U}\,\psi} \stackrel{\text{def}}{=} \langle\cdot\varnothing\cdot\rangle\,(\overline{\varphi}\,\mathbf{U}\,\overline{\psi})$$
$$\overline{\varphi \vee \psi} \stackrel{\text{def}}{=} \overline{\varphi} \vee \overline{\psi} \qquad \overline{\mathsf{E}\varphi} \stackrel{\text{def}}{=} \langle\cdot\mathsf{Agt}\cdot\rangle\,\overline{\varphi}$$

We clearly have $q \models \Phi$ iff $q \models_\varnothing \overline{\Phi}$. The existential path quantifier is replaced by $\langle\cdot\mathsf{Agt}\cdot\rangle$. Now given a path formula $\mathbf{X}\,\varphi$, its corresponding $\overline{\mathbf{X}\,\varphi}$ will be interpreted within a strategy context – fixed by some $\langle\cdot\mathsf{Agt}\cdot\rangle$ modality – defining the path. Thus inserting a $\langle\cdot\varnothing\cdot\rangle$ does not change the semantics and allows us to put $\mathbf{X}$ in the immediate scope of a strategy quantifier as required in $\mathsf{ATL}_{sc,\infty}$. Nevertheless $\overline{\Psi}$ does not yet belong to $\mathsf{ATL}_{sc,\infty}$ because it is possible to have a double embedding of strategy quantifiers of the following form: $\langle\cdot\mathsf{Agt}\cdot\rangle\,(\,\langle\cdot\varnothing\cdot\rangle\,\varphi \wedge \ldots)$. This can be removed by using the method for translating CTL^+into CTL [8] (remember that $\langle\cdot\mathsf{Agt}\cdot\rangle$ is equivalent to E). □

A.5 Comparison with other formalisms

Game Logic. Game Logic was introduced in [2], it extends of $\mathsf{ATL}^\star$ and allows us to deal explicitly with the execution tree induced by a strategy. Let ℓ be a state and F a strategy for some coalition A, we write $\mathsf{ExecTree}(\ell, F)$ for the subtree of the computation tree from ℓ whose infinite rooted paths are the elements of $\mathsf{Out}(\ell, F)$.

Definition 15. *The syntax of GL is defined by the following grammar :*

$$\begin{aligned}\mathsf{GL} \ni \varphi_s, \psi_s &::= P \mid \neg\varphi_s \mid \varphi_s \vee \psi_s \mid \exists A.\varphi_t\\ \varphi_t, \psi_t &::= \varphi_s \mid \neg\varphi_t \mid \varphi_t \vee \psi_t \mid \exists\varphi_p\\ \varphi_p, \psi_p &::= \varphi_t \mid \neg\varphi_p \mid \varphi_p \vee \psi_p \mid \mathbf{X}\,\varphi_p \mid \varphi_p\,\mathbf{U}\,\psi_p\end{aligned}$$

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

Thus in GL, we distinguish state, tree and path formulae. Note also that path formulae are interpreted over a path ρ inside a tree t (denoted (t, ρ)). We give the semantics of the unusual operators :

$$\begin{aligned}\ell \models \exists A.\varphi_t \quad &\text{iff} \quad \exists F_A \in \mathsf{Strat}(A).\ \mathsf{ExecTree}(\ell, F_A) \models \varphi_t,\\ t \models \varphi_s \quad &\text{iff} \quad \ell \models \varphi_s, \text{ where } \ell \text{ is the root of } t,\\ t \models \exists\varphi_p \quad &\text{iff} \quad \exists\rho \in t.\ (t,\rho) \models \varphi_p,\\ (t,\rho) \models \varphi_t \quad &\text{iff} \quad t \models \varphi_t.\end{aligned}$$

Proposition 16. *Every GL formula can be translated into an equivalent* $\mathsf{ATL}^{\star}_{sc}$ *formula.*

Proof. The translation is given by the inductively-defined application σ s.t.:

$$\overline{\exists A.\varphi} \stackrel{\text{def}}{=} \langle\!\langle A \rangle\!\rangle \, \overline{\varphi}, \quad \overline{\exists \varphi} \stackrel{\text{def}}{=} \neg \langle\!\langle \varnothing \rangle\!\rangle \neg \overline{\varphi}, \quad \overline{P} \stackrel{\text{def}}{=} P.$$

The other inductive rules are defined the natural way.

Note that if φ is a GL tree formula, then $\overline{\varphi}$ is an $\mathsf{ATL}^{\star}_{sc}$ state formula. In this translation, we use a strategy context to represent the tree used to interpret GL path and tree formulae. We must show that for any GL path (resp. tree) formula φ_p (resp. φ_t), any path ρ in some CGS and any strategy F for some coalition A, we have: $(\mathsf{ExecTree}(\rho[0], F), \rho) \models \varphi_p$ iff $\rho \models_F \overline{\varphi_p}$ and $\mathsf{ExecTree}(\rho[0], F) \models \varphi_t$ iff $\rho[0] \models_F \overline{\varphi_t}$. Here we just consider the first equivalence (the second one can be treated in a similar war). The usual induction steps are straightforward, thus we only consider the two following cases :

- $\varphi = \exists A\psi$. Then $\rho \models_F \overline{\varphi}$ means that there exists a strategy F' for the coalition A, such that any computation ρ' in $\mathsf{Out}(\rho[0], F')$ will satisfy $\overline{\psi}$. With the induction hypothesis, this is equivalent to $\exists F' \in \mathsf{Strat}(A).\ \forall \rho' \in \mathsf{Out}(\rho[0], F').\ (\mathsf{ExecTree}(\rho[0], F'), \rho') \models \psi$, and so to $\rho[0] \models \exists A.\psi$ because ψ is a tree formula. Now $\exists A\psi$ is a state formula which can be interpreted over any execution tree with root $\rho[0]$, most accordingly over $\mathsf{ExecTree}(\rho[0], F)$.
- $\varphi = \exists \psi$. Then $\rho \models_F \overline{\varphi}$ means that "not all the computations from $\rho[0]$ and following the strategy context F do not satisfy $\overline{\psi}$", and then this is equivalent to $\exists \rho' \in \mathsf{Out}(\rho[0], F)$. $\rho' \models_F \overline{\psi}$. Again, we obtain by applying the induction hypothesis that there exists a path in $\mathsf{ExecTree}(\rho[0], F)$ that satisfies ψ, and then $\mathsf{ExecTree}(\rho[0], F) \models \exists \psi$ and this is equivalent to $(\mathsf{ExecTree}(\rho[0], F), \rho) \models \varphi$ (as φ is a tree formula).

As a result, if we restrict this equivalence to the cases where φ is a state formula and F is the empty strategy, we get that $\overline{\varphi}$ is an $\mathsf{ATL}^{\star}_{sc}$ equivalent formula for φ. □

Proposition 17. $\langle\!\langle A_1 \rangle\!\rangle \, \mathbf{X} \, (\, \langle\!\langle A_2 \rangle\!\rangle \, \mathbf{X} \, b \wedge \langle\!\langle A_3 \rangle\!\rangle \, \mathbf{X} \, a)$ *has no equivalent in GL.*

Proof. Consider the CGSs $\mathcal{S}_1$ and $\mathcal{S}_2$ in Figure 4. They satisfy the same GL formulae, since the third move for Player 1 does not affect the sets of execution trees induced by all strategies for a fixed coalition : for any coalition A and state q, we have $\mathsf{ExecTree}(q, \mathsf{Strat}_{\mathcal{S}_1}(A)) = \mathsf{ExecTree}(q, \mathsf{Strat}_{\mathcal{S}_2}(A))$. Yet this move ensures that s'_0 satisfies $\langle\!\langle A_1 \rangle\!\rangle \, \mathbf{X} \, (\, \langle\!\langle A_2 \rangle\!\rangle \, \mathbf{X} \, b \wedge \langle\!\langle A_3 \rangle\!\rangle \, \mathbf{X} \, a)$ (when players 2 and 3 respectively choose moves 2 and 1), while s_0 does not. □

A.6 PSPACE-hardness of $\mathsf{ATL}_{sc,0}$ model checking

Lemma 18. *The model checking problem for* $\mathsf{ATL}_{sc,0}$ *over CGSs is* PSPACE*-hard.*

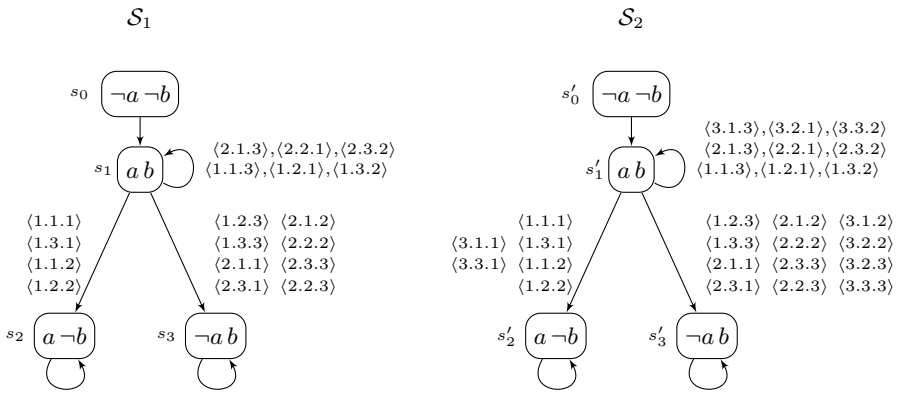

Fig. 4. $\mathcal{S}_1$ and $\mathcal{S}_2$ cannot be distinguished by GL

Proof. We reduce it to the PSPACE-complete problem QBFSAT:

QBFSAT:

Input: A family of variables $X = \{x^1, ..., x^n\}$, a boolean CNF formula $\varphi = \bigwedge_{j=1,...,J} C_j$ on the set of variables X.
Output: The value of $\exists x_1 \forall x_2 \exists x_3 ... Q_n x_n \varphi(x_1, ..., x_n)$, where Q_n is a $\forall$ if n is even, and a $\exists$ otherwise

From an instance $\mathcal{I}$ of this problem, we build the turn-based CGS $\mathcal{C}$ on Fig. 5. In $\mathcal{C}$ we have n different agents, one for each variable, who play only once and so give a value to their variable. In fact the players have successively the choice between two moves, for true or false. We then label the states in order to be able to express that every disjunctive clause C_j will eventually be set to true:

- $\forall 1 \leq i \leq n, \mathsf{Lab}(x_i) = \{C_j \mid x_i \text{ appears in } C_j\}$,
- $\forall 1 \leq i \leq n, \mathsf{Lab}(\neg x_i) = \{C_j \mid \neg x_i \text{ appears in } C_j\}$.

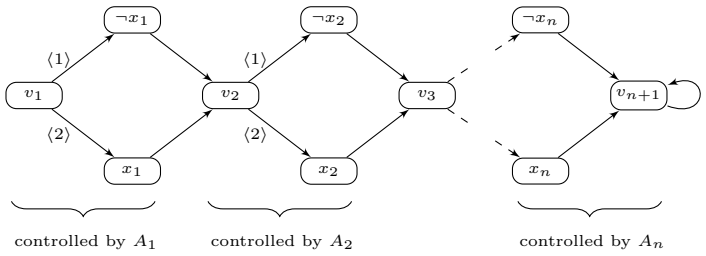

Fig. 5. $\mathcal{C}$ without the labels

Since each player plays once, a strategy for A_i corresponds to a truth value for the variable x_i. Thus, the formula $\langle\!\langle A_i^0 \rangle\!\rangle \psi_s$ (resp. $\neg \langle\!\langle A_i^0 \rangle\!\rangle \neg \psi_s$) means that there exists a truth value for x_i such that (resp. no matter what the truth value for x_i is) ψ_s stands. Note that when a player makes his move, every opponent knows

the move, because it is written in the context. Therefore, those who play after him can take it into consideration when comes their turn.

Now consider the following $\mathsf{ATL}^{\star}_{sc,0}$ formula :

$$\Psi_{\mathcal{I}} \stackrel{\text{def}}{=} \langle\!\cdot A^0_1 \cdot\!\rangle \, (\neg \, \langle\!\cdot A^0_2 \cdot\!\rangle \, \neg (\, \langle\!\cdot A^0_3 \cdot\!\rangle \, (\neg \ldots \epsilon_n \bigwedge_{j=1,\ldots,J} \langle\!\cdot \varnothing^0 \cdot\!\rangle \, \mathbf{F} \, C_j \ldots)))$$

where ϵ_n is $\neg$ if n is even, and $\varnothing$ otherwise. Clearly $\Psi_{\mathcal{I}}$ holds for $\mathcal{C}$ iff $\mathcal{I}$ is a positive instance of QBFSAT. Finally it is easy to obtain an equivalent model checking problem for $\mathsf{ATL}_{sc,0}$. Let $\Psi'_{\mathcal{I}}$ be the $\mathsf{ATL}_{sc,0}$ formula defined from $\Psi_{\mathcal{I}}$ by inserting $\mathbf{X}$ after every $\langle\!\cdot A^0_i \cdot\!\rangle$ modality. And let $\mathcal{C}'$ be the CGS defined from $\mathcal{C}$ by adding n states $q_1, \ldots, q_n$ and $n-1$ transitions $q_i \rightarrow q_{i+1}$ for $i < n$, and the transition $q_n \rightarrow v_1$. Clearly $q_1 \models \Psi'_{\mathcal{I}}$ iff $\mathcal{I}$ is a positive instance. □

It should be noticed that $\cdot\rangle \cdot \langle\cdot$ has not been used in the hardness proof.