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 makes 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, especially we provide a PSPACE algorithm 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 [13, 7, 14]. 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 (ATL). Over the last ten years, ATL has been proposed as a new flavor of temporal logics for specifying and verifying properties in *multi-agent systems* (modeled as *Concurrent Game Structures* (CGS) [2]), in which several agents can concurrently act upon the behaviour of the system. In 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* in two directions: first, while ATL strategy quantifiers drop strategies introduced by earlier quantifiers in the evaluation of the formula, our logics keep executing those strategies. To achieve this idea, we naturally adapt the semantics of ATL* in order to interpret a formula within a *stategy context*. Our new strategy quantifier, written $\langle\cdot A\cdot\rangle$, can for instance express that "A has a strategy s.t. (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: $\langle\cdot A\cdot\rangle\,\mathbf{G}\,\big(\,\langle\cdot B\cdot\rangle\,\Phi\wedge\,\langle\cdot C\cdot\rangle\,\Psi\big)$. Naive attempts to express this property in standard ATL fail: 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; in $\langle\!\langle A\rangle\!\rangle\,\mathbf{G}\,(\,\langle\!\langle A,B\rangle\!\rangle\,\Phi\wedge\,\langle\!\langle A,C\rangle\!\rangle\,\Psi)$, coalition A is allowed to use different strategies when playing with B and C.

Our second extension consists in parameterising 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: on the one hand, we study the increase in expressiveness brought by our extensions, comparing our logics to ATL and ATL* and several related logics such as Game Logic [2], Strategy Logic [6] and $\mathrm{Q}D_\mu$ [12], ... We also illustrate their convenience with some sample formulas expressing *e.g.* equilibrium properties.

On the other hand, we study the model-checking problem for our extensions: while we only have a non-elementary algorithm for the most general logic, 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 work. Recently, several works have focused on the same kind of extensions of ATL, and come up with different solutions which we list below. Generally speaking, this leads to very expressive logics, able to express for instance equilibrium properties, and drastically increases the model-checking complexity.

- IATL [1] extends ATL with strategy contexts, with a similar definition as ours, but it requires 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.
- $\mathrm{Q}D_\mu$ [12] 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 [3] 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.

Instead of defining a completely new formalism, we prefer sticking to an ATL-like syntax, as we believe that our new modality $\langle\cdot\cdot\rangle$ is more intuitive than the standard ATL modality $\langle\langle A\rangle\rangle$. Also, none of the above the extension has the ability to explicitely restrict to bounded-memory strategies, which is of obvious practical relevance and leads to more efficient algorithms.

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. For lack of space, we refer to the full version [4] of this paper for the detailled proofs.

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. In the whole paper, AP denotes a finite non-empty set of atomic propositions.

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* $(\mathsf{Loc}, \ell_0, \mathsf{Lab}, \delta, \mathsf{Agt}, \mathcal{M}, \mathsf{Mov}, \mathsf{Edg})$ *where:*

- Loc *is a finite set of* locations*,* $\ell_0 \in \mathsf{Loc}$ *is the initial location;*
- $\mathsf{Lab}\colon \mathsf{Loc} \to 2^{AP}$ *is a labelling function;*
- $\delta \subseteq \mathsf{Loc} \times \mathsf{Loc}$ *is the set of transitions;*
- $\mathsf{Agt} = \{A_1, ..., A_k\}$ *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.*
- $\mathsf{Edg}\colon \mathsf{Loc} \times \mathcal{M}^k \to \delta$*, where* $k = |\mathsf{Agt}|$*, is a transition table. With each location and each set of moves of the agents, it associates the resulting transition.*

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 of his possible moves m_{A_i} and the next transition

[1] Our results would still hold if we consider symbolic CGSs [10], where the transition table is encoded through boolean formulas.

is given by $\mathsf{Edg}(\ell, m_{A_1}, ..., m_{A_k})$. We write $\mathsf{Next}(\ell)$ for the set of all transitions corresponding to possible moves 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 possible transitions from ℓ when player A_j makes the move m.

2.2 Coalitions, bounded-memory strategies, outcomes.

Coalitions. 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 set of possible moves for coalition A from ℓ. Those moves m are composed of one single move per agent of the coalition, *i.e.*, $m = (m_a)_{a \in A}$.
- 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 $\rho = \ell_0 \ell_1 \dots$ 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 $\rho[i...j]$ for part of ρ between ℓ_i and ℓ_j. In particular, $\rho[i]$ denotes 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 $f_{A_i}(\ell_0 \dots \ell_m) \in \mathsf{Mov}(\ell_m, A_i)$. A strategy is *memoryless* if it only depends on the current state (*i.e.*, $f_{A_i}(\ell_0 \dots \ell_m) = f_{A_i}(\ell_m)$). 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: $\ell_0 \ell_1 \dots \in \mathsf{Out}(\ell, f_{A_j})$ iff $\ell_0 = \ell$ and $\ell_{i+1} \in \mathsf{Next}(\ell_i, A_j, f_{A_j}(\ell_0 \dots \ell_i))$ 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: $\ell_0 \ell_1 \dots \in \mathsf{Out}(\ell, F_A)$ iff $\ell_0 = \ell$ and for any i, $\ell_{i+1} \in \mathsf{Next}(\ell_i, A, (f_{A_j}(\ell_0, \dots, \ell_i))_{A_j \in A})$. 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.

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: $(F \circ F')_{|A_j}(\ell_0 \dots \ell_m)$ equals $F_{|A_j}(\ell_0 \dots \ell_m)$ if $A_j \in A$, and it equals $F'_{|A_j}(\ell_0 \dots \ell_m)$ if $A_j \in B \setminus A$.

Finally, given a strategy F, an execution ρ and some integer $i \geq 0$, we define the strategy $F^{\rho,i}$ corresponding to the behaviour of F after prefix $\rho[0...i]$ as follows: $F^{\rho,i}(\pi) = F(\rho[0 \dots i] \cdot \pi)$. Note that if F is memoryless, then $F^{\rho,i} = F$.

Bounded-memory strategies. Between general strategies (without bound over its resources) and *memoryless* strategies, we can consider *bounded-memory strategies*. Let s be a (binary-encoded) integer representing the size of the memory. We define a bounded memory strategy as a memoryless strategy over the locations of the CGS **and** a set of memory cells [11, 16]: 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. For the sake of readability, given a bounded-memory strategy $F_A = (F^{\mathsf{mov}}, F^{\mathsf{cell}}, c)$, we still write $F_A(\ell)$ for $F^{\mathsf{mov}}(c, \ell)$.

The notions of computations and outcomes are easily extended to this new setting: 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:*

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

with $P \in \mathsf{AP}$, $A \subseteq \mathsf{Agt}$ and $\overline{s} \in (\mathbb{N} \cup \{\infty\})^A$. Formulas defined as φ_s are called state formulas*, while φ_p defines* path formulas*.*

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:

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

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] $\ell_0 \models_\varnothing \Phi$.

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 $\langle\!\cdot A \cdot\rangle\!\rangle$ to denote the modality with no restriction over the memory allowed for the strategies of A (*i.e.*, the modality $\langle\!\cdot A, \infty^A \cdot\rangle\!\rangle$); and we use $\langle\!\cdot A^0 \cdot\rangle\!\rangle$ as an abbreviation for $\langle\!\cdot A, 0^A \cdot\rangle\!\rangle$ to consider only memoryless strategies.

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

Obviously 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, 15]).

We will use standard abbreviations such as $\top = P \vee \neg P$, $\bot = \neg\top$, $\mathbf{F}\, \varphi = \top\, \mathbf{U}\, \varphi$, etc.

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

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

3 Expressiveness

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

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

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 [4], we show how our logics can express the different properties that motivated the introduction of SL, $\mathrm{Q}D_\mu$ or IATL. Here we just give a few examples.

Nash equilibria. Given two players A_1 and A_2 having their own objectives Φ_1 and Φ_2, two strategies F_1 and F_2 for players 1 and 2 respectively form a *Nash equilibrium* 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*. Given a strategy context $F = (F_1, F_2)$, the following formula holds in state ℓ under F iff F_1 and F_2 form a Nash equilibrium in ℓ:

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

This provides us with a way of expressing the existence of Nash equilibria having extra properties.

Winning secure equilibria. The *winning secure equilibrium* [5] (WSE) is a stronger notion of equilibrium: two strategies F_1 and F_2, for players 1 and 2 with objectives Φ_1 and Φ_2 respectively, form a WSE if each player has no better strategy for himself, and no worse strategy for his opponent. Again, the strategy context F is a winning secure equilibrium in ℓ iff the following formula holds in ℓ within F:

$$\begin{aligned}&(\langle\!\langle A_1 \rangle\!\rangle \Phi_1) \Rightarrow \Phi_1 \ \wedge \ (\langle\!\langle A_2 \rangle\!\rangle \Phi_2) \Rightarrow \Phi_2 \ \wedge \\ &\qquad \Big(\langle\!\langle A_1 \rangle\!\rangle (\Phi_1 \wedge \neg\Phi_2) \Rightarrow (\Phi_1 \wedge \neg\Phi_2) \Big) \ \wedge \ \Big(\langle\!\langle A_2 \rangle\!\rangle (\Phi_2 \wedge \neg\Phi_1) \Rightarrow (\Phi_2 \wedge \neg\Phi_1) \Big)\end{aligned}$$

Client-server intreactions. Given a protocol where a server S has to treat the requests of differents agents A_1, ..., A_n, we can express that S has a strategy to ensure that every agent A_i can act in order to make its requests to be granted. Such a property can be stated as folows:

$$\langle\!\langle S \rangle\!\rangle \, \mathbf{G} \Big[\bigwedge_{i=1\ldots n} \Big(\mathsf{req}_i \Rightarrow \langle\!\langle A_i \rangle\!\rangle \, \mathbf{F} \, \mathsf{grant}_i \Big) \Big]$$

Clearly this property requires the use of strategy contexts because every agent has to cooperate with the server (but not with other agents).

3.2 Expressiveness of $\rangle\cdot A\cdot\langle$ quantifier

We have illustrated the use of modality $\rangle\cdot A\cdot\langle$ by expressing the classical $\mathsf{ATL}^\star$ modality $\langle\!\langle A\rangle\!\rangle$ with $\rangle\mathsf{Agt}\langle\cdot\ \langle\cdot A\rangle$: 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 $\rangle\cdot\langle$ 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 (in this definition, $[\cdot C\cdot]\,\varphi \stackrel{\text{def}}{=} \neg\,\langle\cdot C\cdot\rangle\,\neg\varphi$):

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

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

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

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

Now we have the following lemma:

Lemma 4. *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:*

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

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.

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

We have the following equivalences : $\ell \models_F \langle\cdot A,\overline{s}\cdot\rangle\ [\cdot C'\cdot]\,\overline{\varphi}^{C'}$ 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 also

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

since we have

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

This leads to $\ell \models_{F_{\setminus C}} \langle\!\cdot A, \overline{s} \cdot\rangle\!\rangle \varphi$, which is the desired result.

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

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

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

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

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.

We can now finish the proof by considerng $\Psi = \overline{\Phi}^{\varnothing}$. □

3.3 Comparison with other formalisms

Figure 1 summarizes the expressiveness results for our logics. An arrow $L \to L'$ denotes that $L \leq_{ex} L'$, *i.e.*, that L' is at least as expressive as L (*i.e.*, for any formula in L, there exists an equivalent[3] formula in L'). Note that in some cases, the relation is strict and we have $L < L'$, see the corresponding theorems for more details. The dotted arrows correspond to results proved in (the long version of) this paper; plain arrows correspond to literature results (they are labeled with bibliographic references) or direct syntactic inclusions.

The results about ATL, $\mathsf{ATL}^{\star}$, $\mathsf{CTL}^{\star}$ and AMC are presented in [4]: most of them are based on the ability of the new modalities with strategy contexts and/or memory bounds to *distinguish* models that are alternating-bisimilar (and thus satisfy the same formulas of the classical AMC fragments). The full version also contains the proof that adding the quantification over bounded memory increases the expressive power of ATL_{sc} and $\mathsf{ATL}^{\star}_{sc}$.

Here we only develop our results concerning Game Logic, which is a powerful logic to handle properties over strategies, and we discuss the case of Strategy Logic.

[3] That is, having the same truth value in any location of any CGS under any context.

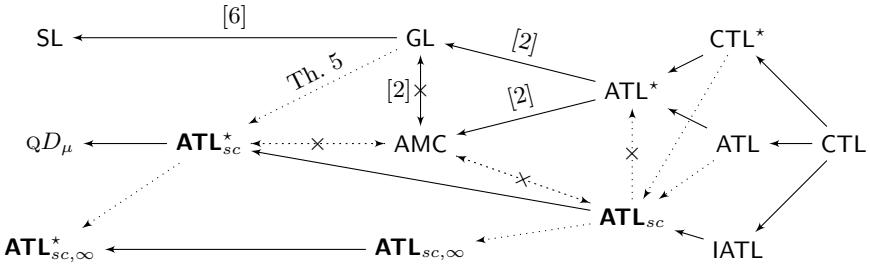

Fig. 1. Expressiveness of $\mathsf{ATL}_{sc,\infty}$ and $\mathsf{ATL}^{\star}_{sc,\infty}$ compared to classical logics

Comparison with Game Logic. *Game Logic* was introduced in [2] in order to express the module-checking problem [9]. This logic is an extension of $\mathsf{ATL}^{\star}$ where the existence of a strategy for A is written $\exists A$, and where it is possible to deal explicitly with the execution tree induced by a strategy: given such a tree t, it is possible to quantify (with modalities $\exists$ and $\forall$) 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 the existence of a strategy F_A for A s.t. in the tree induced by F_A, we have: (1) there exists a run along which $P \,\mathbf{U}\, P'$ holds and (2) every run satisfies $\mathbf{F}\, P''$. We have the following result:

Theorem 5. *$ATL^{\star}_{sc}$ $>_{ex}$ GL*

Proof (sketch). First we give a translation from GL into $\mathsf{ATL}^{\star}_{sc}$; given a GL formula φ, we inductively define $\overline{\varphi}$:

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

The other inductive rules are defined in 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. In the following, given a state ℓ of a CGS and a strategy F for some coalition A, we use $\mathsf{ExecTree}(\ell, F)$ to denote the subtree of the computation tree from ℓ whose infinite rooted paths are the elements of $\mathsf{Out}(\ell, F)$. 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 way). The usual induction steps are straightforward, thus we only consider the following two cases :

- $\varphi = \exists A.\psi$. Then $\rho \models_F \overline{\varphi}$ means that there is a strategy F' for coalition A, s.t. any computation ρ' in $\mathsf{Out}(\rho[0], F')$ satisfies $\overline{\psi}$. By i.h., this is equivalent to $\exists F' \in \mathsf{Strat}(A).\ \forall \rho' \in \mathsf{Out}(\rho[0], F').\ (\mathsf{ExecTree}(\rho[0], F'), \rho') \models \psi$, hence to $\rho[0] \models \exists A.\psi$ because ψ is a tree formula. Now $\exists A.\psi$ is a state formula that can be interpreted over any execution tree with root $\rho[0]$, in particular 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 is then equivalent to $\exists \rho' \in \mathsf{Out}(\rho[0], F)$. $\rho' \models_F \overline{\psi}$. Again from the i.h. we obtain the existence of a path in $\mathsf{ExecTree}(\rho[0], F)$ satisfying ψ, and then $\mathsf{ExecTree}(\rho[0], F) \models \exists\psi$, which is equivalent to $(\mathsf{ExecTree}(\rho[0], F), \rho) \models \varphi$ (as φ is a tree formula).

Finally if we consider the case 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.

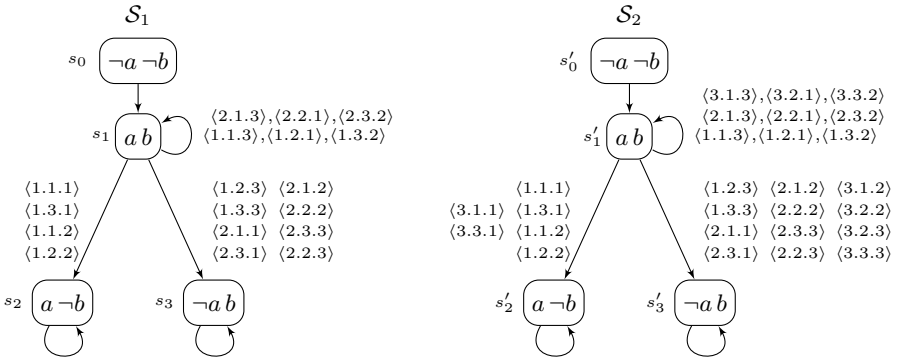

Fig. 2. $\mathcal{S}_1$ and $\mathcal{S}_2$ cannot be distinguished by GL

We have $\mathsf{GL} <_{ex} \mathsf{ATL}^{\star}_{sc}$ because 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. Indeed consider the CGSs $\mathcal{S}_1$ and $\mathcal{S}_2$ in Figure 2. They satisfy the same GL formulas, since move 3 for Player 1 (in $\mathcal{S}_2$) 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. □

Comparison with Strategy Logic [6]. Strategy Logic has 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 (*i.e.*, the strategies for the two players) given to φ indicate on which paths 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 (proving those results seems to be especially challenging due to the particular syntax of SL):

– 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 $\mathsf{ATL}^{\star}_{sc,\infty}$. Indeed this fragment is built from formulas of the form $\exists x_1.\ \exists y_1.\ \forall x_2.\ \forall y_2.\ \big[\varphi_1(x_1,y_2)\wedge\varphi_2(x_2,y_1)\wedge\varphi_3(x_1,y_1)\big]$ [6] which we can express as $\langle\!\cdot A_1\cdot\!\rangle\,\big[\varphi_1\wedge\langle\!\cdot A_2\cdot\!\rangle\,(\varphi_3\wedge\,\cdot\!\rangle A_1\langle\!\cdot\,\varphi_2)\big]$.

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 is non-elementary. We thus mainly focus on simpler cases (namely, memoryless and bounded-memory strategies), where more efficient algorithms can be obtained.

Theorem 6. *Model checking $\mathsf{ATL}^{\star}_{sc,\infty}$ formulas over CGS is decidable.*

Proof (sketch). Our logic $\mathsf{ATL}^{\star}_{sc}$ can be translated into $\mathrm{Q}D_\mu$ (see [4] for more details). This 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 7. *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 similar to that of [3] (and is also detailled in [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 $\langle\!\cdot A\cdot\!\rangle\,\psi$), 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 $\psi = \langle B^0 \rangle \psi'$:
      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 $\psi = \rangle B \langle \psi'$:
      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 $(\mathcal{C}', l_0, \mathbf{A}\sigma(\varphi))$

Remark 1. Note that PSPACE-completeness straightforwardly extends 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 [8, 10]. Also notice that both the above algorithm and the PSPACE-hardness proof can be adapted to 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 Section 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: $\mathbb{M}^s_{A_i} = (\mathsf{Agt}, \mathsf{Loc}^i_s, \varnothing, \mathsf{Loc}^i_s \times \mathsf{Loc}^i_s, \varnothing, \mathcal{M}^i_s \cup \{\bot\}, \mathsf{Mov}^i_s, \mathsf{Edg}^i_s)$ 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-bound vector. Now considering the product structure $\mathcal{C}_{\overline{s}} = \prod_{A_i \in \mathsf{Agt}} \mathbb{M}^{\overline{s}(A_i)}_{A_i} \times \mathcal{C}$, for all players 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 $\mathsf{RStrat}^0_{\mathcal{C}_{\overline{s}}}(A_j) \subseteq \mathsf{Strat}^0_{\mathcal{C}_{\overline{s}}}(A_j)$. Adapting the proof of Theorem 7 to memory-bounded strategies, we get:

Proposition 8. *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 7 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 formulas 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 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 [15].
- 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 finding reasonably efficient algorithms for fragments of these expressive logics. Finally we think that the ability to deal explicitly with bounded-memory strategies is an interesting approach to develop.

References

1. Thomas Ågotnes, Valentin Goranko, and Wojciech 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, June 2007.
2. Rajeev Alur, Thomas A. Henzinger, and Orna Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49(5):672–713, 2002.

3. Christel Baier, Tomáš Brázdil, Marcus Größer, and Antonín 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.
4. Thomas Brihaye, Arnaud Da Costa, François Laroussinie, and Nicolas Markey. ATL with strategy contexts and bounded memory. Technical Report LSV-08-14, Lab. Specification et Verification, February 2008.
5. Krishnendu Chatterjee, Thomas A. Henzinger, and Marcin Jurdziński. Games with secure equilibria. *Theoretical Computer Science*, 365(1-2):67–82, 2006.
6. Krishnendu Chatterjee, Thomas A. Henzinger, and Nir Piterman. Strategy logic. In *Proceedings of the 18th International Conference on Concurrency Theory (CONCUR'07)*, LNCS, pages 59–73. Springer-Verlag, September 2007.
7. Edmund M. Clarke and E. Allen Emerson. Design and synthesis of synchronous skeletons using branching-time temporal logic. In *Proceedings of the 3rd Workshop on Logics of Programs (LOP'81)*, volume 131 of *LNCS*, pages 52–71. Springer, 1981.
8. Wojciech Jamroga and Jürgen 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.
9. Orna Kupferman, Moshe Y. Vardi, and Pierre Wolper. Module checking. *Information and Computation*, 164(2):322–344, January 2001.
10. François Laroussinie, Nicolas Markey, and Ghassan Oreiby. On the expressiveness and complexity of ATL. In *Proceedings of the 10th International Conference on Foundations of Software Science and Computation Structures (FoSSaCS'07)*, volume 4423 of *LNCS*, pages 243–257, Braga, Portugal, 2007. Springer.
11. René Mazala. Infinite games. In *Automata, Logics, and Infinite Games*, volume 2500 of *LNCS*, pages 23–42. Springer-Verlag, 2002.
12. Sophie 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)*, volume 4762 of *LNCS*, pages 253–267. Springer-Verlag, October 2007.
13. Amir Pnueli. The temporal logic of programs. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science (FOCS'77)*, pages 46–57. IEEE Comp. Soc. Press, October-November 1977.
14. Jean-Pierre Queille and Joseph Sifakis. Specification and verification of concurrent systems in CESAR. In *Proceedings of the 5th International Symposium on Programming (SOP'82)*, volume 137 of *LNCS*, pages 337–351. Springer-Verlag, April 1982.
15. Pierre-Yves 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.
16. Wolfgang Thomas. On the synthesis of strategies in infinite games. In *Proceedings of the 12th Symposium on Theoretical Aspects of Computer Science (STACS'95)*, volume 900 of *LNCS*, pages 1–13. Springer-Verlag, March 1995.