Reasoning about Knowledge and Strategies: Epistemic Strategy Logic

Francesco Belardinelli
Laboratoire IBISC – Unversité d'Evry
belardinelli@ibisc.fr

In this paper we introduce Epistemic Strategy Logic (ESL), an extension of Strategy Logic with modal operators for individual knowledge. This enhanced framework allows us to represent explicitly and to reason about the knowledge agents have of their own and other agents' strategies. We provide a semantics to ESL in terms of epistemic concurrent game models, and consider the corresponding model checking problem. We show that the complexity of model checking ESL is not worse than (non-epistemic) Strategy Logic.

1 Introduction

Formal languages to represent and reason about strategies and coalitions are a thriving area of research in Artificial Intelligence and multi-agent system [4, 8, 20]. Recently, a wealth of multi-modal logics have appeared, which allow to formalise complex strategic abilities and behaviours of individual agents and groups [2, 5]. In parallel to these developments, in knowledge representation there is a well-established tradition of extending logics for reactive systems with epistemic operators to reason about the knowledge agents have of systems evolution. These investigations began in the '80s with contributions on combinations of linear- and branching-time temporal logics with multi-agent epistemic languages [9, 10, 6]. Along this line of research, [11] introduced alternating-time temporal epistemic logic (ATEL), an extension of ATL with modalities for individual knowledge. The various flavours of logics of time and knowledge have been successfully applied to the specification of distributed and multi-agent systems in domains as diverse as security protocols, UAVs, web services, and e-commerce, as well as to verification by model checking [7, 16].

In this paper we take inspiration from the works above and pursue further this line of research by introducing Epistemic Strategy Logic, an extension of Strategy Logic (SL) [5, 17] that allows agents to reason about their strategic abilities. The extension here proposed is naive in the sense that it suffers many of the shortcomings of its relative ATEL [12]. Nonetheless, we reckon that it constitutes an excellent starting point to analyse the interaction of knowledge and strategic abilities in a language, such as SL, that explicitly allow for quantification on strategies.

Related Work. This paper builds on previous contributions on Strategy Logic. SL has been introduced in [5] for two-player concurrent game structures (CGS). In [17] the semantics has been extended to a multi-player setting. Also, [17] introduced bind operators for strategies in the syntax. In the present contribution we consider multi-agent CGS in line with [17]. However, we adopt an agent-based perspective and consider agents with possibly different actions and protocols [6]. Also, our language do not include bind operators to avoid the formal machinery associated with these operators. We leave such an extension for future and more comprehensive work. Finally, the model checking results in Section 4 are inspired by and use techniques from [17].

Submitted to:
SR2014

Even though to our knowledge no epistemic extension of SL has been proposed yet, the interaction between knowledge and strategic reasoning has been studied extensively, especially in the context of alternating-time temporal logic. An extension of ATL with knowledge operators, called ATEL, was put forward in [11], and immediately imperfect information variants of this logic were considered in [14], which introduces alternating-time temporal observational logic (ATOL) and ATEL-R*, as well as uniform strategies. Notice that [14] also analyses the distinction between *de re* and *de dicto* knowledge of strategies; this distinction will also be considered later on in the context of Epistemic Strategy Logic. Further, [13] enriches ATL with a constructive notion of knowledge. As regards (non-epistemic) ATL, more elaborate notions of strategy have been considered. In [1] commitment in strategies has been analysed; while [15] introduced a notion of "feasible" strategy. In future work it might be worth exploring to what extent the theoretical results available for the various flavours of ATEL transfer to ESL.

Scheme of the paper. In Section 2 we introduce the epistemic concurrent game models (ECGM), which are used in Section 3 to provide a semantics to Epistemic Strategy Logic (ESL). In Section 4 we consider the model checking problem for this setting and state the corresponding complexity results. Finally, in Section 5 we discuss the results and point to future research. For reasons of space, all proofs are omitted. An extended version of this paper with complete proofs is available [3].

2 Epistemic Concurrent Game Models

In this section we present the epistemic concurrent game models (ECGM), an extension of concurrent game structures [2, 11], starting with the notion of *agent*.

Definition 1 (Agent) *An* agent *is a tuple* $i = \langle L_i, Act_i, Pr_i \rangle$ *such that*

- L_i *is the set of* local states $l_i, l'_i, \ldots$*;*
- Act_i *is the finite set of* actions $\sigma_i, \sigma'_i, \ldots$*;*
- $Pr_i : L_i \mapsto 2^{Act_i}$ *is the* protocol function.

Intuitively, each agent i is situated in some local state $l_i \in L_i$, representing her local information, and performs the actions in Act_i according to the protocol function Pr_i [6]. Differently from [17], we assume that agents have possibly different actions and protocols. To formally describe the interactions between agents, we introduce their synchronous composition. Given a set AP of atomic propositions and a set $Ag = \{i_0, \ldots, i_n\}$ of agents, we define the set L of global states $s, s', \ldots$ (resp. the set Act of joint actions $\sigma, \sigma', \ldots$) as the cartesian product $L_0 \times \ldots \times L_n$ (resp. $Act_0 \times \ldots \times Act_n$). In what follows we denote the jth component of a tuple t as t_j or, equivalently, as $t(j)$.

Definition 2 (ECGM) *Given a set* $Ag = \{i_0, \ldots, i_n\}$ *of agents* $i = \langle L_i, Act_i, Pr_i \rangle$*, an* epistemic concurrent game model *is a tuple* $\mathscr{P} = \langle Ag, s_0, \tau, \pi \rangle$ *such that*

- $s_0 \in L$ *is the* initial global state*;*
- $\tau : L \times Act \mapsto L$ *is the* global transition function*, where* $\tau(s, \sigma)$ *is defined iff* $\sigma_i \in Pr_i(l_i)$ *for every* $i \in Ag$*;*
- $\pi : AP \mapsto 2^L$ *is the* interpretation function *for atomic propositions in AP.*

The transition function τ describes the evolution of the ECGM from the initial state s_0. We now introduce some notation that will be used in the rest of the paper. The *transition relation* $\to$ on global states is defined as $s \to s'$ iff there exists $\sigma \in Act$ s.t. $\tau(s, \sigma) = s'$. A *run* λ from a state s, or s-run, is an infinite sequence $s^0 \to s^1 \to \ldots$, where $s^0 = s$. For $n, m \in \mathbb{N}$, with $n \leq m$, we define $\lambda(n) = s^n$

and $\lambda[n,m] = s^n, s^{n+1}, \dots, s^m$. A state s' is *reachable from* s if there exists an s-run λ s.t. $\lambda(i) = s'$ for some $i \geq 0$. We define S as the set of states reachable from the initial state s_0. Further, let $\sharp$ be a placeholder for arbitrary individual actions. Given a subset $A \subseteq Ag$ of agents, an *A-action* σ_A is an $|Ag|$-tuple s.t. *(i)* $\sigma_A(i) \in Act_i$ for $i \in A$, and *(ii)* $\sigma_A(j) = \sharp$ for $j \notin A$. Then, Act_A is the set of all A-actions and $D_A(s) = \{\sigma_A \in Act_A \mid \text{for every } i \in A, \sigma_i \in Pr_i(l_i)\}$ is the set of all A-actions enabled at $s = \langle l_0, \dots, l_n \rangle$. A joint action σ *extends* an A-action σ_A, or $\sigma_A \sqsubseteq \sigma$, iff $\sigma_A(i) = \sigma(i)$ for all $i \in A$. The *outcome* $out(s, \sigma_A)$ *of action* σ_A *at state* s is the set of all states s' s.t. there exists a joint action $\sigma \sqsupseteq \sigma_A$ and $\tau(s, \sigma) = s'$. Finally, two global states $s = \langle l_0, \dots, l_n \rangle$ and $s' = \langle l'_0, \dots, l'_n \rangle$ are *indistinguishable* for agent i, or $s \sim_i s'$, iff $l_i = l'_i$ [6].

3 Epistemic Strategy Logic

We now introduce Epistemic Strategy Logic as a specification language for ECGM. Hereafter we consider a set Var_i of strategy variables $x_i, x'_i, \dots$, for every agent $i \in Ag$.

Definition 3 (ESL) *For $p \in AP$, $i \in Ag$ and $x_i \in Var_i$, the ESL formulas ϕ are defined in BNF as follows:*

$$\phi \quad ::= \quad p \mid \neg\phi \mid \phi \to \phi \mid X\phi \mid \phi U \phi \mid K_i\phi \mid \exists x_i \phi$$

The language ESL is an extension of the Strategy Logic in [5] to a multi-agent setting, including an epistemic operator K_i for each $i \in Ag$. Alternatively, ESL can be seen as the epistemic extension of the Strategy Logic in [17], minus the bind operator. We do not consider bind operators in ESL for ease of presentation. The ESL formula $\exists x_i \phi$ is read as "agent i has some strategy to achieve ϕ". The interpretation of LTL operators X and U is standard. The epistemic formula $K_i\phi$ intuitively means that "agent i knows ϕ". The other propositional connectives and LTL operators, as well as the strategy operator $\forall$, can be defined as standard. Also, notice that we can introduce the *nested-goal* fragment ESL[NG], the *boolean-goal* fragment ESL[BG], and the *one-goal* fragment ESL[1G] in analogy to SL [17]. Further, the *free* variables $fr(\phi) \subseteq Ag$ of an ESL formula ϕ are inductively defined as follows:

$$\begin{array}{lcl}
fr(p) & = & \emptyset \\
fr(\neg\phi) = fr(K_i\phi) & = & fr(\phi) \\
fr(\phi \to \phi') & = & fr(\phi) \cup fr(\phi') \\
fr(X\phi) = fr(\phi U \phi') & = & Ag \\
fr(\exists x_i \phi) & = & fr(\phi) \setminus \{i\}
\end{array}$$

A *sentence* is a formula ϕ with $fr(\phi) = \emptyset$, and the set $bnd(\phi)$ of bound variables is defined as $Ag \setminus fr(\phi)$.

To provide a semantics to ESL formulas in terms of ECGM, we introduce the notion of strategy.

Definition 4 (Strategy) *Let γ be an ordinal s.t. $1 \leq \gamma \leq \omega$ and $A \subseteq Ag$ a set of agents. A* γ-recall A-strategy *is a function $F_A[\gamma] : \bigcup_{1 \leq n < 1+\gamma} S^n \mapsto \bigcup_{s \in S} D_A(s)$ s.t. $F_A[\gamma](\kappa) \in D_A(last(\kappa))$ for every $\kappa \in \bigcup_{1 \leq n < 1+\gamma} S^n$, where $1 + \gamma = \gamma$ for $\gamma = \omega$ and $last(\kappa)$ is the last element of κ.*

Hence, a γ-recall A-strategy returns an enabled A-action for every sequence $\kappa \in \bigcup_{1 \leq n < 1+\gamma} S^n$ of states of length at most γ. Notice that for $A = \{i\}$, $F_A[\gamma]$ can be seen as a function from $\bigcup_{1 \leq n < 1+\gamma} S^n$ to Act_i s.t. $F_A[\gamma](\kappa) \in Pr_i(last(\kappa))$ for $\kappa \in \bigcup_{1 \leq n < 1+\gamma} S^n$. In what follows we write $F_i[\gamma]$ for $F_{\{i\}}[\gamma]$. Then, for $A = \{i_0, \dots, i_m\} \subseteq Ag$, $F_A[\gamma]$ is equal to $F_{i_0}[\gamma] \times \ldots \times F_{i_m}[\gamma]$, where for every $\kappa \in \bigcup_{1 \leq n < 1+\gamma} S^n$, $(F_{i_0}[\gamma] \times \ldots \times F_{i_m}[\gamma])(\kappa)$ is defined as the set of actions $\sigma \in \bigcup_{s \in S} D_A(s)$ s.t. $\sigma_i = F_i[\gamma](\kappa)$ if $i \in A$, $\sigma_i = \sharp$ otherwise. Therefore, a group strategy is the composition of its members' strategies. Further, the *outcome* of strategy $F_A[\gamma]$ at state s, or $out(s, F_A[\gamma])$, is the set of all s-runs λ s.t. $\lambda(i+1) \in out(\lambda(i), F[\gamma](\lambda[j,i]))$ for all $i \geq 0$ and $j = max(i - \gamma + 1, 0)$. Depending on γ we can define positional strategies, strategies with perfect recall, etc. [8]. However, these different choices do not affect the following results, so we assume that γ is fixed and omit it. Moreover, by Def. 4 it is apparent that agents have perfect information, as their strategies are determined by global states [4]; we leave contexts of imperfect information for future research.

Now let χ be an assignment that maps each agent $i \in Ag$ to an i-strategy F_i. For $Ag = \{i_0, \dots, i_n\}$, we denote $\chi(i_0) \times \ldots \times \chi(i_n)$ as F^{χ}, that is, the Ag-strategy s.t. for every $\kappa \in \bigcup_{1 \leq n < 1+\gamma} S^n$, $F^{\chi}(\kappa) = \sigma \in \bigcup_{s \in S} D_{Ag}(s)$ iff $\sigma_i = \chi(i)(\kappa)$ for every $i \in Ag$. Since $|out(s, F^{\chi})| = 1$, we simply write $\lambda = out(s, F^{\chi})$. Also, $\chi^i_{F_i}$ denotes the assignment s.t. *(i)* for all agents j different from i, $\chi^i_{F_i}(j) = \chi(j)$, and *(ii)* $\chi^i_{F_i}(i) = F_i$.

Definition 5 (Semantics of ESL) *We define whether an ECGM* $\mathscr{P}$ satisfies *a formula* φ *at state* s *according to assignment* χ*, or* $(\mathscr{P}, s, \chi) \models \varphi$*, as follows (clauses for propositional connectives are straightforward and thus omitted):*

$(\mathscr{P}, s, \chi) \models p$ *iff* $s \in \pi(p)$
$(\mathscr{P}, s, \chi) \models X\psi$ *iff for* $\lambda = out(s, F^{\chi})$, $(\mathscr{P}, \lambda(1), \chi) \models \psi$
$(\mathscr{P}, s, \chi) \models \psi U \psi'$ *iff for* $\lambda = out(s, F^{\chi})$ *there is* $k \geq 0$ *s.t.* $(\mathscr{P}, \lambda(k), \chi) \models \psi'$ *and* $0 \leq j < k$ *implies* $(\mathscr{P}, \lambda(j), \chi) \models \psi$
$(\mathscr{P}, s, \chi) \models K_i\psi$ *iff for all* $s \in S$, $s \sim_i s'$ *implies* $(\mathscr{P}, s', \chi) \models \psi$
$(\mathscr{P}, s, \chi) \models \exists x_i \psi$ *iff there exists an i-strategy* F_i *s.t.* $(\mathscr{P}, s, \chi^i_{F_i}) \models \psi$

An ESL formula φ is *satisfied* at state s, or $(\mathscr{P}, s) \models \varphi$, if $(\mathscr{P}, s, \chi) \models \varphi$ for all assignments χ; φ is *true* in $\mathscr{P}$, or $\mathscr{P} \models \varphi$, if $(\mathscr{P}, s_0) \models \varphi$. The satisfaction of formulas is independent from bound variables, that is, $\chi(fr(\phi)) = \chi'(fr(\phi))$ implies that $(\mathscr{P}, s, \chi) \models \phi$ iff $(\mathscr{P}, s, \chi') \models \phi$. In particular, the satisfaction of sentences is independent from assignments.

We can now state the model checking problem for ESL.

Definition 6 (Model Checking Problem) *Given an ECGM* $\mathscr{P}$ *and an ESL formula* ϕ*, determine whether there exists an assignment* χ *s.t.* $(\mathscr{P}, s_0, \chi) \models \phi$*.*

Notice that, if $y_1, \dots, y_m$ is an enumeration of $fr(\phi)$, then the model checking problem amounts to check whether $\mathscr{P} \models \exists y_1, \dots, \exists y_m \phi$, where $\exists y_1, \dots, \exists y_m \phi$ is a sentence.

Hereafter we illustrate the formal machinery introduced thus far with a toy example.

Example. We introduce a turn-based ECGM with two agents, A and B. First, A secretly chooses between 0 and 1. Then, at the successive stage, B also chooses between 0 and 1. The game is won by agent A if the values provided by the two agents coincide, otherwise B wins. We formally describe this toy game starting with agents A and B. Specifically, A is the tuple $\langle L_A, Act_A, Pr_A \rangle$, where *(i)* $L_A = \{\varepsilon_A, 0, 1\}$; *(ii)* $Act_A = \{set(0), set(1), skip\}$; and *(iii)* $Pr_A(\varepsilon_A) = \{set(0), set(1)\}$ and $Pr_A(0) = Pr_A(1) = \{skip\}$. Further, agent B is defined as the tuple $\langle L_B, Act_B, Pr_B \rangle$, where $L_B = \{\varepsilon_B, \lambda, 0, 1\}$; $Act_B = \{wait, set(0), set(1), skip\}$; $Pr_B(\varepsilon_B) = \{wait\}$, $Pr_B(\lambda) = \{set(0), set(1)\}$ and $Pr_B(0) = Pr_B(1) = \{skip\}$. The intuitive meaning of local states, actions and protocol functions is clear. Also, we consider the set $AP = \{win_A, win_B\}$ of atomic propositions, which intuitively express that agent A (resp. B) has won the game. We now introduce the ECGM $\mathscr{Q}$, corresponding to our toy game, as the tuple $\langle Ag, s_0, \tau, \pi \rangle$, where *(i)* $s_0 = (\varepsilon_A, \varepsilon_B)$; *(ii)* the transition function τ is given as follows for $i, j \in \{0, 1\}$:

- $\tau((\varepsilon_A, \varepsilon_B), (set(i), wait)) = (i, \lambda)$
- $\tau((i, \lambda), (skip, set(j))) = (i, j)$
- $\tau((i, j), (skip, skip)) = (\varepsilon_A, \varepsilon_B)$

and *(iii)* $\pi(win_A) = \{(0,0), (1,1)\}$, $\pi(win_B) = \{(1,0), (0,1)\}$. Notice that we suppose that our toy game, represented in Fig. 1, is non-terminating.

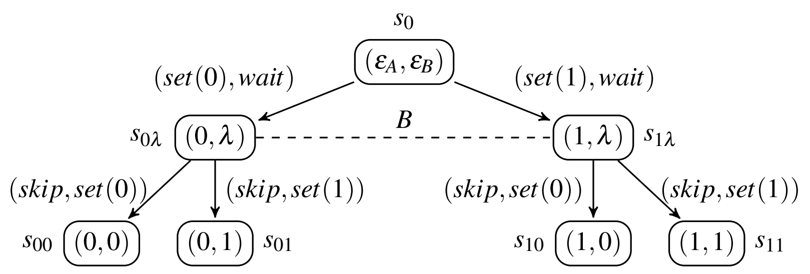

Figure 1: the ECGM $\mathcal{Q}$. Transitions from s_{00}, s_{01}, s_{10}, and s_{11} to s_0 are omitted.

Now, we check whether the following ESL specifications hold in the ECGM $\mathcal{Q}$.

$$\mathcal{Q} \models \forall x_A\, X\, K_B\, \exists y_B\, X\, win_B \tag{1}$$

$$\mathcal{Q} \not\models \forall x_A\, X\, \exists y_B\, K_B\, X\, win_B \tag{2}$$

$$\mathcal{Q} \models \forall x_A\, X\, K_B\, K_A\, \exists y_B\, X\, win_A \tag{3}$$

$$\mathcal{Q} \models \forall x_A\, X\, K_B\, \exists y_B\, K_A\, X\, win_A \tag{4}$$

Intuitively, (1) expresses the fact that at the beginning of the game, independently from agent A's move, at the next step agent B knows that there exists a move by which she can enforce her victory. That is, if agent A chose 0 (resp. 1), then B can choose 1 (resp. 0). However, B only knows that there exists a move, but she is not able to point it out. In fact, (2) does not hold, as B does not know which specific move A chose, so she is not capable of distinguishing states $s_{0\lambda}$ and $s_{1\lambda}$. Moreover, by (3) B knows that A knows that there exists a move by which B can let A win. Also, by (4) this move is known to A, as it is the B-move matching A's move.

Indeed, in ESL it is possible to express the difference between *de re* and *de dicto* knowledge of strategies. One of the first contributions to tackle this issue formally is [14]. Formula (1) expresses agent B's *de dicto* knowledge of strategy y_B; while (2) asserts *de re* knowledge of the same strategy. Similarly, in (3) agent A has *de re* knowledge of strategy y_B; while (4) states that agent A knows the same strategy *de dicto*. The *de re*/*de dicto* distinction is of utmost importance as, as shown above, having a *de dicto* knowledge of a strategy does not guarantee that an agent is actually capable of performing the associated sequence of actions. Ideally, in order to have an effective strategy, agents must know it *de re*.

4 Model Checking ESL

In this section we consider the complexity of the model checking problem for ESL. For an ESL formula ϕ we define $alt(\phi)$ as the maximum number of alternations of quantifiers $\exists$ and $\forall$ in ϕ. Then, ESL[k-alt] is the set of ESL formulas ϕ with $alt(\phi)$ equal to or less than k. We can now state the hardness result for ESL.

Theorem 1 (Hardness) *The model checking problem for ESL[k-alt] is k-EXPSPACE-hard.*

This result is inspired to Theorem 3.5 in [17], as its proof reduces the satisfiability problem for the quantified propositional temporal logic QPTL to ESL model checking. However, the reduction provided in Appendix A is different, as [17] makes use of the bind operator, which is not available in ESL.

Finally, we state the following completeness result, corresponding to Theorem 5.8 in [17].

Theorem 2 (Completeness) *The model checking problem for ESL is PTIME-complete w.r.t. the size of the model and NON-ELEMENTARYTIME w.r.t. the size of the formula.*

We provide a proof in Appendix B and observe that for the nested-goal fragment ESL[NG] it is possible to show that the model checking problem is in $(k+1)$-EXPTIME w.r.t. the maximum alternation k of a formula. We conclude that the complexity of model checking ESL is not worse than the corresponding problem for the Strategy Logic in [17].

5 Conclusions

In this paper we introduced Epistemic Strategy Logic, an extension of Strategy Logic [17] with modalities for individual knowledge. We provided this specification language with a semantics in terms of epistemic concurrent game models (ECGM), and analysed the corresponding model checking problem. A number of developments for the proposed framework are possible. Firstly, the model checking problem for the nested-goal, boolean-goal, and one-goal fragment of SL has lower complexity. It is likely that similar results hold also for the corresponding fragments of ESL. Secondly, we can extend ESL with modalities for group knowledge, such as common and distributed knowledge. Thirdly, we can consider various assumptions on ECGM, for instance perfect recall, no learning, and synchronicity. The latter two extensions, while enhancing the expressive power of the logic, are also likely to increase the complexity of the model checking and satisfiability problems.

References

[1] Thomas Agotnes, Valentin Goranko & Wojciech Jamroga (2007): *Alternating-time Temporal Logics with Irrevocable Strategies*. In: *Proceedings of the 11th Conference on Theoretical Aspects of Rationality and Knowledge*, TARK '07, ACM, New York, NY, USA, pp. 15–24, doi:10.1145/1324249.1324256.

[2] Rajeev Alur, Thomas A. Henzinger & Orna Kupferman (2002): *Alternating-time temporal logic*. *J. ACM* 49(5), pp. 672–713, doi:10.1145/585265.585270.

[3] Francesco Belardinelli (2014): *Reasoning about Knowledge and Strategies: Epistemic Strategy Logic*. Technical Report, Universit d'Evry, Laboratoire IBISC. Available at `https://www.ibisc.univ-evry.fr/~belardinelli/Documents/sr2014.pdf`.

[4] Nils Bulling, Jurgen Dix & Wojciech Jamroga (2010): *Model Checking Logics of Strategic Ability: Complexity**. In Mehdi Dastani, Koen V. Hindriks & John-Jules Charles Meyer, editors: *Specification and Verification of Multi-agent Systems*, Springer US, pp. 125–159, doi:10.1007/978-1-4419-6984-2.

[5] Krishnendu Chatterjee, Thomas A. Henzinger & Nir Piterman (2010): *Strategy logic*. *Inf. Comput.* 208(6), pp. 677–693, doi:10.1016/j.ic.2009.07.004.

[6] R. Fagin, J.Y. Halpern, Y. Moses & M.Y. Vardi (1995): *Reasoning About Knowledge*. The MIT Press.

[7] Peter Gammie & Ron van der Meyden (2004): *MCK: Model Checking the Logic of Knowledge*. In Rajeev Alur & Doron Peled, editors: *CAV*, *Lecture Notes in Computer Science* 3114, Springer, pp. 479–483, doi:10.1007/978-3-540-27813-9_41.

[8] Valentin Goranko & Wojciech Jamroga (2004): *Comparing Semantics of Logics for Multi-Agent Systems*. *Synthese* 139(2), pp. 241–280, doi:10.1023/B:SYNT.0000024915.66183.d1.

[9] Joseph Y. Halpern & Moshe Y. Vardi (1986): *The Complexity of Reasoning about Knowledge and Time: Extended Abstract*. In Juris Hartmanis, editor: *STOC*, ACM, pp. 304–315, doi:10.1145/12130.12161.

[10] Joseph Y. Halpern & Moshe Y. Vardi (1989): *The Complexity of Reasoning about Knowledge and Time. I. Lower Bounds*. *J. Comput. Syst. Sci.* 38(1), pp. 195–237, doi:10.1016/0022-0000(89)90039-1.

[11] W. van der Hoek & M. Wooldridge (2003): *Cooperation, Knowledge, and Time: Alternating-time Temporal Epistemic Logic and its Applications*. Studia Logica 75(1), pp. 125–157, doi:10.1023/A:1026185103185.

[12] Wojciech Jamroga (2004): *Some Remarks on Alternating Temporal Epistemic Logic*. In: Proceedings of Formal Approaches to Multi-Agent Systems (FAMAS 2003), pp. 133–140.

[13] Wojciech Jamroga & Thomas Ågotnes (2007): *Constructive knowledge: what agents can achieve under imperfect information*. Journal of Applied Non-Classical Logics 17(4), pp. 423–475, doi:10.3166/jancl.17.423-475.

[14] Wojciech Jamroga & Wiebe van der Hoek (2004): *Agents that Know How to Play*. Fundam. Inform. 63(2-3), pp. 185–219. Available at `http://iospress.metapress.com/content/xh738axb47d8rchf/`.

[15] G. Jonker (2003): *Feasible strategies in Alternating-time Temporal Epistemic Logic*. Master's thesis, University of Utrecht.

[16] A. Lomuscio, H. Qu & F. Raimondi (2009): *MCMAS: A Model Checker for the Verification of Multi-Agent Systems*. In A. Bouajjani & O. Maler, editors: CAV, Lecture Notes in Computer Science 5643, Springer, pp. 682–688, doi:10.1007/978-3-642-02658-4_55.

[17] Fabio Mogavero, Aniello Murano, Giuseppe Perelli & Moshe Y. Vardi (2011): *Reasoning About Strategies: On the Model-Checking Problem*. CoRR abs/1112.6275. Available at `http://arxiv.org/abs/1112.6275`.

[18] D. Muller & P. Schupp (1995): *Simulating Alternating Tree Automata by Nondeterministic Automata: New Results and New Proofs of Theorems of Rabin*. Theoretical Computer Science 141, pp. 69–107.

[19] David E. Muller & Paul E. Schupp (1987): *Alternating Automata on Infinite Trees*. Theor. Comput. Sci. 54, pp. 267–276, doi:10.1016/0304-3975(87)90133-2.

[20] Marc Pauly (2002): *A Modal Logic for Coalitional Power in Games*. J. Log. Comput. 12(1), pp. 149–166, doi:10.1093/logcom/12.1.149.

A Model Checking ESL: lower bound

In this section we prove that model checking ESL formulas is non-elementary-hard. Specifically, we show that for ESL formulas with maximum alternation k the model checking problem is k-EXPSPACE-hard. The proof strategy is similar to [17], namely, we reduce the satisfiability problem for quantified propositional temporal logic (QPTL) to ESL model checking. However, the reduction applied is different, as ESL does not contain the bind operator used in the reduction in [17].

First, we introduce QPTL formulas by the following BNF, for $p \in AP$:

$$\psi \quad ::= \quad p \mid \neg\psi \mid \psi \to \psi \mid X\psi \mid F\psi \mid \exists p\psi$$

In analogy to ESL, we can define the notion of sentence and the fragment QPTL[k-alt] of QPTL formulas with maximum alternation equal to or less than k. To interpreted QPTL formulas we introduce *temporal evaluations* as functions te from $\mathbb{N}$ to the set $\{\mathtt{t}, \mathtt{f}\}$ of truth values. Further, a *propositional evaluation pe* is a function from AP to the set TE of temporal evalutations. Similarly to strategy assignments, the propositional evaluation pe^{p}_{te} assigns te to p and coincides with pe on all other atomic propositions. We can now define the notion of satisfaction for QPTL.

Definition 7 (Semantics of QPTL) *We define whether a propositional evaluation pe* satisfies *a QPTL formula φ at time k, or $(pe,k) \models \varphi$, as follows (clauses for propositional connectives are straightforward and thus omitted):*

$(pe,k) \models p$	*iff*	$pe(p)(k) = \mathtt{t}$
$(pe,k) \models X\psi$	*iff*	$(pe,k+1) \models \psi$
$(pe,k) \models F\psi$	*iff*	*there exists* $j \geq k$, $(pe,j) \models \psi$
$(pe,k) \models \exists p\psi$	*iff*	*there exists a temporal evaluation te s.t.* $(pe^{p}_{te},k) \models \psi$

We now prove that the satisfiability problem for QPTL sentences built on a finite set $AP = \{p_0, \ldots, p_n\}$ of atomic propositions can be reduced to model checking ESL sentences on a ECGM $\mathcal{Q}$ of fixed size on $|AP|$, albeit exponential. The main result follows from the next lemma.

Lemma 3 (QPTL Reduction) *Let $AP = \{p_0, \ldots, p_n\}$ be a finite set of atomic propositions. There exists an ECGM $\mathcal{Q}$ on AP s.t. for every QPTL[k-alt] sentence ϕ on AP, there exists an ESL[k-alt] sentence $\overline{\phi}$ s.t. ϕ is satisfiable iff $\mathcal{Q} \models \overline{\phi}$.*

Sketch of Proof. First, we introduce an agent $p = \langle L_p, Act_p, Pr_p \rangle$ for every atomic proposition $p \in AP$, defined as follows:

- $L_p = \{\top, \bot\}$;
- $Act_p = \{\mathfrak{t}, \mathfrak{f}\}$ is the set of truth values;
- $Pr_p(l) = Act_p$ for every $l \in L_p$.

Then, we introduce the ECGM $\mathcal{Q} = \langle Ag, I, \tau, \pi \rangle$ where

- $s_0 = \langle \bot, \ldots, \bot \rangle \in L_{p_0} \times \ldots \times L_{p_n}$ for definiteness;
- $\tau(s, \sigma) = s'$ where for every $p \in Ag$, $s'_p = \top$ (resp. $\bot$) iff $\sigma_p = \mathfrak{t}$ (resp. $\mathfrak{f}$);
- $s \in \pi(p)$ iff $s_p = \top$.

Further, we define a translation function $\overline{}$ from QPTL to ESL formulas:

$$\begin{array}{lcl} \overline{q} & = & Xq \\ \overline{\flat\phi} & = & \flat\overline{\phi} \\ \overline{\phi \star \phi'} & = & \overline{\phi} \star \overline{\phi'} \\ \overline{\exists q \phi} & = & \exists x_q \overline{\phi} \end{array}$$

where $\flat$ (resp. $\star$) is any unary (resp. binary) operator.

It is easy to check that for every QPTL formula ϕ, $alt(\phi) = alt(\overline{\phi})$. Hence, $\overline{}$ is indeed a translation from QPTL[k-alt] to ESL[k-alt].

We can now prove the following result from which it follows that a QPTL[k-alt] sentence ϕ is satisfiable iff the ESL[k-alt] sentence $\overline{\phi}$ is true in $\mathcal{Q}$.

Lemma 4 *For every QPTL formula ϕ, $(pe, i) \models \phi$ iff $(\mathcal{Q}, \lambda(i), \chi) \models \overline{\phi}$, where $\lambda = out(s_0, F^{\chi})$ and $\chi(p)(\lambda[0,n]) = pe(p)(n)$ for every $n \in \mathbb{N}$.*

Sketch of Proof. For $\phi = p$, $(pe, i) \models \phi$ iff $pe(p)(i) = \mathbf{t}$. Hence, by definition $\chi(p)(\lambda[0,i]) = \mathbf{t}$ for $\lambda = out(s_0, F^{\chi})$. Also, we observe that $(\mathcal{Q}, \lambda(i), \chi) \models Xp$ iff for $\lambda' = out(\lambda(i), F^{\chi})$, $(\mathcal{Q}, \lambda'(1), \chi) \models p$, iff $\lambda'(1)(p) = \top$. In particular, this means that $\chi(p)(\lambda'(0)) = \mathbf{t}$. We observe that $\lambda'(0) = \lambda(i)$ and $\chi(p)(\lambda'(0)) = \chi(p)(\lambda[0,i]) = \mathfrak{t}$. As a result, $(pe, i) \models p$ iff $pe(p)(i) = \chi(p)(\lambda[0,i]) = \mathfrak{t} = \chi(p)(\lambda'(0))$, iff $(\mathcal{Q}, \lambda'(1), \chi) \models p$, iff $(\mathcal{Q}, \lambda(i), \chi) \models Xp$.

The inductive cases for propositional connectives and LTL operators are straightforward.

Let $\phi = \exists p \psi$. $\Rightarrow$ If $(pe, i) \models \phi$ then there exists te s.t. $(pe^p_{te}, i) \models \psi$. Now consider a p-strategy F_p s.t. for all $n \in \mathbb{N}$, $F_p(\lambda[0,n]) = te(n)$. Then, for all $n \in \mathbb{N}$, $\chi^p_{F_p}(q)(\lambda[0,n]) = pe^p_{te}(q)(n)$ for all $q \in AP$. Therefore, by induction hypothesis we have that $(\mathcal{Q}, \lambda(i), \chi^p_{F_p}) \models \overline{\psi}$, that is, $(\mathcal{Q}, \lambda(i), \chi) \models \overline{\phi}$.

$\Leftarrow$ If $(\mathcal{Q}, \lambda(i), \chi) \models \overline{\phi}$ then there exists a p-strategy F_p s.t. $(\mathcal{Q}, \lambda(i), \chi^p_{F_p}) \models \overline{\psi}$. Now consider the temporal evaluation te s.t. for all $n \in \mathbb{N}$, $te(n) = F_p(\lambda[0,n])$. As above, for all $n \in \mathbb{N}$, $pe^p_{te}(q)(n) = \chi^p_{F_p}(q)(\lambda[0,n])$ for all $q \in AP$. Therefore, by induction hypothesis we obtain that $(pe^p_{te}, i) \models \psi$, that is, $(pe, i) \models \phi$. $\square$

By Lemma 4 we complete the proof of Lemma 3. In particular, QPTL satisfiability is reducible to ESL model checking. □

By this result and the fact that the satisfiability problem for QPTL[k-alt] is k-EXPSPACE-hard [17], we can derive Theorem 1.

Theorem 1 (Hardness) *The model checking problem for ESL[k-alt] is k-EXPSPACE-hard.*

In particular, it follows that ESL model checking is non-elementary-hard.

B Model Checking ESL: upper bound

In this section we extend to Epistemic Strategy Logic the model checking procedure for SL in [17], which is based on alternating tree automata [19]. First, let Δ be a finite set of directions $\{d_0, \ldots, d_h\}$. Further, for some set U, U^+ is the set of finite, non-empty, sequences of elements in U.

Definition 8 (Tree) *A Δ-tree is a set $T \subseteq \Delta^+$ s.t.* (i) *if $x \cdot j \in T$ for $x \in \Delta^+$ and $j \in \Delta$, then $x \in T$;* (ii) *there is only one $j \in \Delta$ in T (i.e., the root).*

The sequences $x \in \Delta^+$ in T are called *nodes*. For $x \in T$, $j \in \Delta$, the nodes $x \cdot j$ are the *successors* of x. A *leaf* is a node with no successors.

Definition 9 (Branch) *A* branch *in a Δ-tree T is a non-empty set $B \subseteq T$ such that* (i) *$root_T \in B$, and* (ii) *for every $x \in \Delta^+$ in B, either x is a leaf or there exists a unique $j \in \Delta$ such that $x \cdot j \in B$.*

Given an alphabet Σ, a Σ-labelled tree is a pair $\mathscr{T} = \langle T, V \rangle$ where *(i)* T is a tree, and *(ii)* $V : T \to \Sigma$ maps each node of T to a letter in Σ.

We now introduce alternating tree automata. In what follows $\mathscr{B}^+(AP)$ is the set of positive Boolean formulas over a set AP of atomic propositions. For instance, $\mathscr{B}^+(\{p,q\})$ includes $p \wedge q$, $p \vee q$, $p \wedge p$.

Definition 10 (ATA) *An* alternating tree automaton *is a tuple $\mathscr{A} = \langle \Sigma, \Delta, Q, q_0, \delta, \mathscr{F} \rangle$ where*

- Σ *is a finite* alphabet*;*
- Δ *is a finite set of* directions *as above;*
- Q *is a finite set of* states*;*
- $q_0 \in Q$ *is the* initial state*;*
- $\delta : Q \times \Sigma \to \mathscr{B}^+(\Delta \times Q)$ *is the* transition function*;*
- $\mathscr{F} = (F_1, \ldots, F_k) \in (2^Q)^+$ *with $F_1 \subseteq \ldots \subseteq F_k = Q$.*

Intuitively, when the automaton is in state q and reads a node that is labelled by σ, it applies the transition $\delta(q, \sigma)$. Then, a run is the execution of an ATA over a tree.

Definition 11 (Run) *A* run *of an ATA $\mathscr{A}$ over a Σ-labelled Δ-tree $\mathscr{T} = \langle T, V \rangle$ is a $(\Delta \times Q)$-tree T_r such that*

(i) if d is the root of T, then (d, q_0) is the root of T_r;

(ii) for $x = \prod_{i=1}^{n}(d_i, q_i) \in T_r$ and $y = \prod_{i=1}^{n} d_i \in T$, if $\delta(q_n, V(y)) = \theta \in \mathscr{B}^+(\Delta \times Q)$ then there is a (possibly empty) set $S \subseteq \Delta \times Q$ such that

(a) the assignment that assigns $\mathtt{t}$ *to all the atomic propositions in S satisfies θ;*

(b) *for every $(d,q) \in S$ we have $x \cdot (d,q) \in T_r$.*

Notice that if, for some node y, the transition function δ returns the value $\mathfrak{t}$, then y may have no successor. Also, δ can never return the value $\mathfrak{f}$ in a run. We use the term *run* to designate different notions in ECGM and ATA, in order to be consistent with [6, 17]. Also, σ may refer to actions in ECGM or symbols in Σ. The context will disambiguate. A run T_r is *accepting* if all its infinite branches satisfy the acceptance condition. Here we consider a parity acceptance condition. Given a run T_r and an infinite branch $B \subseteq T_r$, let $inf(B) \subseteq Q$ be the set of $q \in Q$ such that there are infinitely many $x \in B$ for which $last(x) \in (\Delta \times \{q\})$, that is, $inf(B)$ contains exactly all the states that appear infinitely often in B. The acceptance condition is then defined as follows:

- A branch B satisfies the parity condition iff the least index i for which $inf(B) \cap F_i \neq \emptyset$ is even.

An automaton *accepts* a tree if and only if there exists a run that accepts it. We denote by $\mathscr{L}(\mathscr{A})$ the set of all Σ-labelled trees that $\mathscr{A}$ accepts. In what follows we consider only ATA coupled with a parity acceptance condition. So, we refer simply to ATA for brevity.

We now show how ATA can be used to decide the model checking problem for ESL. First, we remark that ECGM can be encoded as particular labelled trees. In what follows we take the set Δ of directions as $S \times (Ag \cup \{t\})$. Further, a *temporal epistemic run* λ from a state s, or t.e. s-run, is an infinite sequence $s^0 \rightsquigarrow s^1 \rightsquigarrow \ldots$, where $s^0 = s$ and either $s^i \rightarrow s^{i+1}$ or $s^i \sim_j s^{i+1}$ for some $j \in Ag$.

Definition 12 (Assignment-State Encoding) *Let $\mathscr{P}$ be an ECGM, s a state in $\mathscr{P}$, and χ a strategy assignment. An* assignment-state encoding *for s and χ over $A \subseteq Ag$ is an $(Act_A \times S)$-labelled Δ-tree $\mathscr{T}_s^\chi = \langle T_s^\chi, V_s^\chi \rangle$ such that*

- *T_s^χ is the set of finite sequences $(s^1, j_1), \ldots, (s^n, j_n) \in \Delta^+$ s.t. $s^0, s^1, \ldots, s^n$ is a fragment of a t.e. s-run for $s^0 = s$, and if $s^i \rightarrow s^{i+1}$ then $j_{i+1} = t$, if $s^i \sim_k s^{i+1}$ then $j_{i+1} = k$;*
- *$V(x) = (F_A^\chi(\kappa_{s \cdot x}), last(\kappa_{s \cdot x}))$ where $\kappa_{s \cdot x} \in \bigcup_{1 \leq n < 1+\gamma} S^n$ is the finite fragment of run obtained from $s \cdot x$ by considering the maximal suffix of $x \in \Delta^+$ containing only elements in $S \times \{t\}$, and then eliminating the component t as above.*

We can now prove the following result, which extends Lemma 5.6 in [17]. In what follows $\sigma|_A$ is the restriction of action σ to $A \subseteq Ag$, that is, the A-action s.t. $\sigma|_A(i) = \sigma(i)$ for $i \in A$, and $\sigma|_A(i) = \sharp$ otherwise.

Lemma 5 *Let $\mathscr{P}$ be an ECGM and ϕ an ESL formula. Then, there exists an ATA $\mathscr{A}_{\mathscr{P}}^\phi = \langle Act_{fr(\phi)} \times S, \Delta, Q_\phi, q_{0\phi}, \delta_\phi, \mathscr{F}_\phi \rangle$ s.t. for every state $s \in S$ and assignment χ, we have that $(\mathscr{P}, s, \chi) \models \phi$ iff $\mathscr{T}_s^\chi \in \mathscr{L}(\mathscr{A}_{\mathscr{P}}^\phi)$.*

Sketch of Proof. The result is proved by induction on the structure of ϕ. In the base case of an atomic proposition p, we exhibit an ATA for p. The induction step is perform by applying transformations on ATA corrisponding to each logical operator. The construction is analogous to [17], but in the present case the directions are not just states in S, but couples of states and indexes from $Ag \cup \{t\}$, which represent whether a transition is epistemic or temporal. Hereafter we consider the most significant cases and refer to [17] for further details.

For $\phi = p$, $\mathscr{A}_{\mathscr{P}}^\phi = \langle \{\sigma_\sharp\} \times S, \Delta, \{p\}, p, \delta_p, (\{p\}) \rangle$, where $\sigma_\sharp = \langle \sharp, \ldots, \sharp \rangle \in Act_\emptyset$ and $\delta_p((\sigma_\sharp, s), p) = \mathfrak{t}$ if $s \in \pi(p)$, $\delta_\phi((\sigma_\sharp, s), p) = \mathfrak{f}$ otherwise.

For $\phi = \neg\psi$, $\mathscr{A}_{\mathscr{P}}^\phi$ is the complement automaton of $\mathscr{A}_{\mathscr{P}}^\psi$, as defined in [19] for instance.

For $\phi = \psi_1 \star \psi_2$, where $\star \in \{\wedge, \vee\}$, $\mathscr{A}_{\mathscr{P}}^\phi = \langle Act_{fr(\phi)} \times S, \Delta, Q_\phi, q_{0\phi}, \delta_\phi, \mathscr{F}_\phi \rangle$ is such that

- $Q_\phi = \{q_{0\phi}\} \cup Q_{\psi_1} \cup Q_{\psi_2}$ with $q_{0\phi} \notin Q_{\psi_1} \cup Q_{\psi_2}$

- $\delta_\phi((\sigma,s),q_{0\phi}) = \delta_{\psi_1}((\sigma|_{fr(\psi_1)},s),q_{0\psi_1}) \star \delta_{\psi_2}((\sigma|_{fr(\psi_2)},s),q_{0\psi_2})$
- $\delta_\phi((\sigma,s),q) = \delta_{\psi_1}((\sigma|_{fr(\psi_1)},s),q)$ if $q \in Q_{\psi_1}$, and $\delta_\phi((\sigma,s),q) = \delta_{\psi_2}((\sigma|_{fr(\psi_2)},s),q)$ if $q \in Q_{\psi_2}$
- $\mathscr{F}_\phi = (F_{1\phi},\ldots,F_{k\phi})$, where *(i)* $\mathscr{F}_{\psi_1} = (F_{1\psi_1},\ldots,F_{k_1\psi_1})$ and $\mathscr{F}_{\psi_2} = (F_{1\psi_2},\ldots,F_{k_2\psi_2})$; *(ii)* $k = \max(k_1,k_2)$; *(iii)* $h = \min(k_1,k_2)$; *(iv)* $F_{i\phi} = F_{i\psi_1} \cup F_{i\psi_2}$ for $i \leq h$; *(v)* $F_{i\phi} = F_{i\psi_j}$ for $h+1 \leq i \leq k-1$ and $k_j = k$; and *(vi)* $F_{k\phi} = Q_\phi$.

For $\phi = X\psi$, $\mathscr{A}^\phi_\mathscr{P} = \langle Act \times S, \Delta, Q_\phi, q_{0\phi}, \delta_\phi, \mathscr{F}_\phi\rangle$ is such that

- $Q_\phi = \{q_{0\phi}\} \cup Q_\psi$ with $q_{0\phi} \notin Q_\psi$
- $\delta_\phi((\sigma,s),q_{0\phi}) = ((\tau(s,\sigma),t),q_{0\psi})$ for $\sigma \in Act$
- $\delta_\phi((\sigma,s),q) = \delta_\psi((\sigma|_{fr(\psi)},s),q)$ for all $q \in Q_\psi$
- $\mathscr{F}_\phi = (F_{1\psi},\ldots,F_{k\psi} \cup \{q_{0\phi}\})$.

For $\phi = K_i\psi$, $\mathscr{A}^\phi_\mathscr{P} = \langle Act_{fr(\phi)} \times S, \Delta, Q_\phi, q_{0\phi}, \delta_\phi, \mathscr{F}_\phi\rangle$ is such that

- $Q_\phi = \{q_{0\phi}\} \cup Q_\psi$ with $q_{0\phi} \notin Q_\psi$
- $\delta_\phi((\sigma,s),q_{0\phi}) = \bigwedge_{s' \sim_i s}((s',i),q_{0\psi})$
- $\delta_\phi((\sigma,s),q) = \delta_\psi((\sigma,s),q)$ for all $q \in Q_\psi$
- $\mathscr{F}_\phi = (F_{1\psi},\ldots,F_{k\psi} \cup \{q_{0\phi}\})$.

For $\phi = \exists x_i\psi$, assume that $i \in fr(\psi)$. Then, we can use the operation of existential projection for non-deterministic tree automata. First, we non-determinize the ATA $\mathscr{A}^\psi_\mathscr{P}$ by applying the typical transformation [18], and obtain an equivalent $\mathscr{N}^\psi_\mathscr{P} = \langle Act_{fr(\psi)} \times S, \Delta, Q_\psi, q_{0\psi}, \delta_\psi, \mathscr{F}_\psi\rangle$. Now we define $\mathscr{A}^\phi_\mathscr{P} = \langle Act_{fr(\phi)} \times S, \Delta, Q_\psi, q_{0\psi}, \delta_\phi, \mathscr{F}_\psi\rangle$ where δ_ϕ is such that

- $\delta_\phi((\sigma,s),q) = \bigvee_{\sigma_i \in D_i(s)} \delta_\psi((\sigma^i_{\sigma_i},s),q)$ for all $q \in Q_\psi$

where $\sigma^i_{\sigma_i}$ is the joint action obtained by substituting the ith component $\sigma(i)$ with the individual action σ_i for agent i.

Finally, by induction on the structure of ϕ, we can prove that for every $s \in S$ and assignment χ, $(\mathscr{P},s,\chi) \models \phi$ iff $\mathscr{T}^\chi_s \in \mathscr{L}(\mathscr{A}^\phi_\mathscr{P})$, where $\mathscr{T}^\chi_s$ is the assignment-state encoding for s and χ over $fr(\phi) \subseteq Ag$. □

We now prove the following result that corresponds to Theorem 5.4 in [17].

Theorem 6 (ATA Direction Projection) *Let $\mathscr{A}^\phi_\mathscr{P} = \langle Act_{fr(\phi)} \times S, \Delta, Q_\phi, q_{0\phi}, \delta_\phi, \mathscr{F}_\phi\rangle$ be the ATA obtained in Lemma 5. Moreover, let $s \in S$ be a distinguished state. Then, there exists a non-deterministic ATA $\mathscr{N}^\phi_{\mathscr{P},s} = \langle Act_{fr(\phi)}, \Delta, Q', q'_0, \delta', \mathscr{F}'\rangle$ s.t. for all $Act_{fr(\phi)}$-labelled Δ-tree $\mathscr{T} = \langle T, V\rangle$, we have that $\mathscr{T} \in \mathscr{L}(\mathscr{N}^\phi_{\mathscr{P},s})$ iff $\mathscr{T}' \in \mathscr{L}(\mathscr{A}^\phi_\mathscr{P})$, where $\mathscr{T}'$ is the $(Act_{fr(\phi)} \times S)$-labelled Δ-tree $\langle T, V'\rangle$ s.t. $V'(x) = (V(x), last(\kappa_{s\cdot x}))$, where $\kappa_{s\cdot x}$ is defined as above.*

Sketch of Proof. First we use the non-determinization procedure in [18] and transform the ATA $\mathscr{A}^\phi_\mathscr{P}$ obtained in Lemma 5 into an equivalent non-deterministic ATA $\mathscr{N} = \langle Act_{fr(\phi)} \times S, \Delta, Q'', q''_0, \delta'', \mathscr{F}''\rangle$. Then, we transform $\mathscr{N}$ into the new non-deterministic ATA $\mathscr{N}^\phi_{\mathscr{P},s} = \langle Act_{fr(\phi)}, \Delta, Q', q'_0, \delta', \mathscr{F}'\rangle$, where $Q' = Q'' \times S$, $q'_0 = (q''_0, s)$, $\mathscr{F}' = (F_1 \times S, \ldots, F_k \times S)$ for $\mathscr{F}'' = (F_1,\ldots,F_k)$, and $\delta'(\sigma,(q,s')) = \delta''((\sigma,s'),q))$. Finally, we can check that $\mathscr{N}^\phi_{\mathscr{P},s}$ satisfies the statement of the theorem. □

Notice that for an $Act_{fr(\phi)}$-labelled Δ-tree $\mathscr{T} = \langle T, V\rangle$, we have that the $(Act_{fr(\phi)} \times S)$-labelled Δ-tree $\mathscr{T}' = \langle T, V'\rangle$ in the statement of Theorem 6 is a state-assignment encoding for s and the assignment χ s.t. for every $i \in fr(\phi)$, $\chi(i)(\kappa_{s\cdot x})$ is equal to $V(x)(i)$, for every $x \in T$

Then, by using Lemma 5 and Theorem 6 we can prove the following result.

Theorem 7 *Let $\mathscr{P}$ be an ECGM, s a state in $\mathscr{P}$, χ an assignment, and ϕ an ESL formula. Then, the non-deterministic ATA $\mathscr{N}^{\phi}_{\mathscr{P},s}$ is such that $(\mathscr{P},s,\chi) \models \phi$ iff $\mathscr{L}(\mathscr{N}^{\phi}_{\mathscr{P},s}) \neq \emptyset$.*

Sketch of Proof. $\Rightarrow$ Suppose that $(\mathscr{P},s,\chi) \models \phi$. Then, by Lemma 5 we have that $\mathscr{T}^{\chi}_{s} \in \mathscr{L}(\mathscr{A}^{\phi}_{\mathscr{P}})$, where $\mathscr{T}^{\chi}_{s}$ is the assignment-state encoding for s and χ over $fr(\phi)$. Hence, by Theorem 6 we obtain that $\mathscr{L}(\mathscr{N}^{\phi}_{\mathscr{P},s}) \neq \emptyset$.

$\Leftarrow$ Suppose that $\mathscr{L}(\mathscr{N}^{\phi}_{\mathscr{P},s}) \neq \emptyset$. Then, by Theorem 6 there exists an $(Act_{fr(\phi)} \times S)$-labelled Δ-tree $\mathscr{T}'$ s.t. $\mathscr{T}' \in \mathscr{L}(\mathscr{A}^{\phi}_{\mathscr{P}})$. By reasoning as above, $\mathscr{T}' = \langle T, V' \rangle$, can be seen as a state-assignment encoding for s and the assignment χ s.t. for every $i \in fr(\phi)$, $\chi(i)(\kappa_{s \cdot x})$ is equal to the ith element of the first component of $V'(x)$. Hence, by Lemma 5, we have that $(\mathscr{P},s,\chi) \models \phi$. □

We can finally state the following extension to Theorem 5.8 in [17], which follows from the fact that the non-emptyness problem for alternating tree automata is non-elementary in the size of the formula.

Theorem 2 (Completeness) *The model checking problem for ESL is PTIME-complete w.r.t. the size of the model and NON-ELEMENTARYTIME w.r.t. the size of the formula.*

We conclude by remarking that we can use Theorem 2 to show that the model checking problem for the nested-goal fragment ESL[NG] is PTIME-complete w.r.t. the size of the model and $(k+1)$-EXPTIME w.r.t. the maximum alternation k of a formula.