Reasoning About Strategies:
On the Model-Checking Problem

FABIO MOGAVERO, ANIELLO MURANO, and GIUSEPPE PERELLI,
Universitá degli Studi di Napoli "Federico II", Napoli, Italy.
MOSHE Y. VARDI,
Rice University, Houston, Texas, USA.

In *open systems verification*, to formally check for *reliability*, one needs an appropriate formalism to model the *interaction* between *agents* and express the *correctness* of the system no matter how the *environment* behaves. An important contribution in this context is given by *modal logics* for *strategic ability*, in the setting of *multi-agent games*, such as ATL, ATL*, and the like. Recently, Chatterjee, Henzinger, and Piterman introduced *Strategy Logic*, which we denote here by CHP-SL, with the aim of getting a powerful framework for reasoning explicitly about strategies. CHP-SL is obtained by using *first-order quantifications* over strategies and has been investigated in the very specific setting of *two-agents turned-based games*, where a non-elementary model-checking algorithm has been provided. While CHP-SL is a very expressive logic, we claim that it does not fully capture the strategic aspects of multi-agent systems.

In this paper, we introduce and study a more general strategy logic, denoted SL, for reasoning about strategies in *multi-agent concurrent games*. We prove that SL includes CHP-SL, while maintaining a decidable model-checking problem. In particular, the algorithm we propose is computationally not harder than the best one known for CHP-SL. Moreover, we prove that such a problem for SL is NonElementarySpace-hard. This negative result has spurred us to investigate here syntactic fragments of SL, strictly subsuming ATL*, with the hope of obtaining an elementary model-checking problem. Among the others, we study the sublogics SL[NG], SL[BG], and SL[1G]. They encompass formulas in a special *prenex normal form* having, respectively, nested temporal goals, Boolean combinations of goals and, a single goal at a time. About these logics, we prove that the model-checking problem for SL[1G] is 2ExpTime-complete, thus not harder than the one for ATL*. In contrast, SL[NG] turns out to be NonElementarySpace-hard, strengthening the corresponding result for SL. Finally, we observe that SL[BG] includes CHP-SL, while its model-checking problem relies between NonElementaryTime and 2ExpTime.

1. INTRODUCTION

In system design, *model checking* is a well-established formal method that allows to automatically check for global system correctness [Clarke and Emerson 1981; Queille and Sifakis 1981; Clarke et al. 2002]. In such a framework, in order to check whether a system satisfies a required property, we describe its structure in a mathematical model (such as *Kripke struc-*

tures [Kripke 1963] or *labeled transition systems* [Keller 1976]), specify the property with a formula of a temporal logic (such as LTL [Pnueli 1977], CTL [Clarke and Emerson 1981], or CTL* [Emerson and Halpern 1986]), and check formally that the model satisfies the formula. In the last decade, interest has arisen in analyzing the behavior of individual components or sets of them in systems with several entities. This interest has started in reactive systems, which are systems that interact continually with their environments. In *module checking* [Kupferman et al. 2001], the system is modeled as a module that interacts with its environment and correctness means that a desired property holds with respect to all such interactions.

Starting from the study of module checking, researchers have looked for logics focusing on the strategic behavior of agents in multi-agent systems [Alur et al. 2002; Pauly 2002; Jamroga and van der Hoek 2004]. One of the most important development in this field is *Alternating-Time Temporal Logic* (ATL*, for short), introduced by Alur, Henzinger, and Kupferman [Alur et al. 2002]. ATL* allows reasoning about strategies of agents with temporal goals. Formally, it is obtained as a generalization of CTL* in which the path quantifiers, *there exists* "E" and *for all* "A", are replaced with *strategic modalities* of the form "$\langle\langle A \rangle\rangle$" and "$[\![A]\!]$", where A is a set of *agents* (a.k.a. *players*). Strategic modalities over agent sets are used to express cooperation and competition among them in order to achieve certain goals. In particular, these modalities express selective quantifications over those paths that are the result of infinite games between a coalition and its complement.

ATL* formulas are interpreted over *concurrent game structures* (CGS, for short) [Alur et al. 2002], which model interacting processes. Given a CGS $\mathcal{G}$ and a set A of agents, the ATL* formula $\langle\langle A \rangle\rangle\psi$ is satisfied at a state s of $\mathcal{G}$ if there is a set of strategies for agents in A such that, no matter strategies are executed by agents not in A, the resulting outcome of the interaction in $\mathcal{G}$ satisfies ψ at s. Thus, ATL* can express properties related to the interaction among components, while CTL* can only express property of the global system. As an example, consider the property "processes α and β cooperate to ensure that a system (having more than two processes) never enters a failure state". This can be expressed by the ATL* formula $\langle\langle \{\alpha, \beta\} \rangle\rangle G \neg fail$, where G is the classical LTL temporal operators "*globally*". CTL*, in contrast, cannot express this property [Alur et al. 2002]. Indeed, it can only assert whether the set of all agents may or may not prevent the system from entering a failure state.

The price that one has to pay for the greater expressiveness of ATL* is the increased complexity of model checking. Indeed, both its model-checking and satisfiability problems are 2EXPTIME-COMPLETE [Alur et al. 2002; Schewe 2008].

Despite its powerful expressiveness, ATL* suffers from a strong limitation, due to the fact that strategies are treated only implicitly, through modalities that refer to games between competing coalitions. To overcome this problem, Chatterjee, Henzinger, and Piterman introduced *Strategy Logic* (CHP-SL, for short) [Chatterjee et al. 2007], a logic that treats strategies in *two-player turn-based games* as explicit *first-order objects*. In CHP-SL, the ATL* formula $\langle\langle \{\alpha\} \rangle\rangle\psi$, for a system modeled by a CGS with agents α and β, becomes $\exists x.\forall y.\psi(x, y)$, i.e., "there exists a player-α strategy x such that for all player-β strategies y, the unique infinite path resulting from the two players following the strategies x and y satisfies the property ψ". The explicit treatment of strategies in this logic allows to state many properties not expressible in ATL*. In particular, it is shown in [Chatterjee et al. 2007] that ATL*, in the restricted case of two-agent turn-based games, corresponds to a proper one-alternation fragment of CHP-SL. The authors of that work have also shown that the model-checking problem for CHP-SL is decidable, although only a non-elementary algorithm for it, both in the size of system and formula, has been provided, leaving as open question whether an algorithm with a better complexity exists or not. The complementary question about the decidability of the satisfiability problem for CHP-SL was also left open and, as far as we known, it is not addressed in other papers apart our preliminary work [Mogavero et al. 2010a].

While the basic idea exploited in [Chatterjee et al. 2007] to quantify over strategies and then to commit agents explicitly to certain of these strategies turns to be very powerful and useful [Fisman et al. 2010], CHP-SL still presents severe limitations. Among the others, it needs to

be extended to the more general concurrent multi-agent setting. Also, the specific syntax considered there allows only a weak kind of strategy commitment. For example, CHP-SL does not allow different players to share the same strategy, suggesting that strategies have yet to become first-class objects in this logic. Moreover, an agent cannot change his strategy during a play without forcing the other to do the same.

These considerations, as well as all questions left open about decision problems, led us to introduce and investigate a new *Strategy Logic*, denoted SL, as a more general framework than CHP-SL, for explicit reasoning about strategies in *multi-agent concurrent games*. Syntactically, SL extends LTL by means of two *strategy quantifiers*, the existential $\langle\langle x \rangle\rangle$ and the universal $[[x]]$, as well as *agent binding* (a, x), where a is an agent and x a variable. Intuitively, these elements can be respectively read as *"there exists a strategy x"*, *"for all strategies x"*, and *"bind agent a to the strategy associated with x"*. For example, in a CGS with the three agents α, β, γ, the previous ATL* formula $\langle\langle\{\alpha, \beta\}\rangle\rangle G \neg fail$ can be translated in the SL formula $\langle\langle x \rangle\rangle\langle\langle y \rangle\rangle[[z]](\alpha, x)(\beta, y)(\gamma, z)(G \neg fail)$. The variables x and y are used to select two strategies for the agents α and β, respectively, while z is used to select one for the agent γ such that their composition, after the binding, results in a play where *fail* is never met. Note that we can also require, by means of an appropriate choice of agent bindings, that agents α and β share the same strategy, using the formula $\langle\langle x \rangle\rangle[[z]](\alpha, x)(\beta, x)(\gamma, z)(G \neg fail)$. Furthermore, we may vary the structure of the game by changing the way the quantifiers alternate, as in the formula $\langle\langle x \rangle\rangle[[z]]\langle\langle y \rangle\rangle(\alpha, x)(\beta, y)(\alpha, z)(G \neg fail)$. In this case, x remains uniform w.r.t. z, but y becomes dependent on it. Finally, we can change the strategy that one agent uses during the play without changing those of the other agents, by simply using nested bindings, as in the formula $\langle\langle x \rangle\rangle\langle\langle y \rangle\rangle[[z]]\langle\langle w \rangle\rangle(\alpha, x)(\beta, y)(\gamma, z)(G (\gamma, w)G \neg fail)$. The last examples intuitively show that SL is a extension of both ATL* and CHP-SL. It is worth noting that the pattern of modal quantifications over strategies and binding to agents can be extended to other linear-time temporal logics than LTL, such as the linear μCALCULUS [Vardi 1988]. In fact, the use of LTL here is only a matter of simplicity in presenting our framework, and changing the embedded temporal logic only involves few side-changes in proofs and decision procedures.

As one of the main results in this paper about SL, we show that the model-checking problem is non-elementarily decidable. To gain this, we use an *automata-theoretic approach* [Kupferman et al. 2000]. Precisely, we reduce the decision problem for our logic to the emptiness problem of a suitable *alternating parity tree automaton*, which is an *alternating tree automaton* (see [Grädel et al. 2002], for a survey) along with a *parity acceptance condition* [Muller and Schupp 1995]. Due to the operations of projection required by the elimination of quantifications on strategies, which induce at any step an exponential blow-up, the overall size of the required automaton is non-elementary in the size of the formula, while it is only polynomial in the size of the model. Thus, together with the complexity of the automata-nonemptiness calculation, we obtain that the model checking problem is in PTIME, w.r.t. the size of the model, and NONELEMENTARYTIME, w.r.t. the size of the specification. Hence, the algorithm we propose is computationally not harder than the best one known for CHP-SL and even a non-elementary improvement with respect to the model. This fact allows for practical applications of SL in the field of system verification just as those done for the monadic second-order logic on infinite objects [Elgaard et al. 1998]. Moreover, we prove that our problem has a non-elementary lower bound. Specifically, it is k-EXPSPACE-HARD in the alternation number k of quantifications in the specification.

The contrast between the high complexity of the model-checking problem for our logic and the elementary one for ATL* has spurred us to investigate syntactic fragments of SL, strictly subsuming ATL*, with a better complexity. In particular, by means of these sublogics, we would like to understand why SL is computationally more difficult than ATL*.

The main fragments we study here are *Nested-Goal*, *Boolean-Goal*, and *One-Goal Strategy Logic*, respectively denoted by SL[NG], SL[BG], and SL[1G]. Note that the last, differently from the first two, was introduced in [Mogavero et al. 2012]. They encompass formulas in a special prenex normal form having nested temporal goals, Boolean combinations of goals, and a single goal at a time,

respectively. For goal we mean an SL formula of the type $\flat\psi$, where $\flat$ is a binding prefix of the form $(\alpha_1, x_1), \ldots, (\alpha_n, x_n)$ containing all the involved agents and ψ is an agent-full formula. With more detail, the idea behind SL[NG] is that, when in ψ there is a quantification over a variable, then there are quantifications of all free variables contained in the inner subformulas. So, a subgoal of ψ that has a variable quantified in ψ itself cannot use other variables quantified out of this formula. Thus, goals can be only nested or combined with Boolean and temporal operators. SL[BG] and SL[1G] further restrict the use of goals. In particular, in SL[1G], each temporal formula ψ is prefixed by a quantification-binding prefix $\wp\flat$ that quantifies over a tuple of strategies and binds them to all agents.

As main results about these fragments, we prove that the model-checking problem for SL[1G] is 2EXPTIME-COMPLETE, thus not harder than the one for ATL*. On the contrary, for SL[NG], it is both NONELEMENTARYTIME and NONELEMENTARYSPACE-HARD and thus we enforce the corresponding result for SL. Finally, we observe that SL[BG] includes CHP-SL, while the relative model-checking problem relies between 2EXPTIME and NONELEMENTARYTIME.

To achieve all positive results about SL[1G], we use a fundamental property of the semantics of this logic, called *elementariness*, which allows us to strongly simplify the reasoning about strategies by reducing it to a set of reasonings about actions. This intrinsic characteristic of SL[1G], which unfortunately is not shared by the other fragments, asserts that, in a determined history of the play, the value of an existential quantified strategy depends only on the values of strategies, from which the first depends, on the same history. This means that, to choose an existential strategy, we do not need to know the entire structure of universal strategies, as for SL, but only their values on the histories of interest. Technically, to describe this property, we make use of the machinery of *dependence map*, which defines a Skolemization procedure for SL, inspired by the one in first order logic.

By means of elementariness, we can modify the SL model-checking procedure via alternating tree automata in such a way that we avoid the projection operations by using a dedicated automaton that makes an action quantification for each node of the tree model. Consequently, the resulting automaton is only exponential in the size of the formula, independently from its alternation number. Thus, together with the complexity of the automata-nonemptiness calculation, we get that the model-checking procedure for SL[1G] is 2EXPTIME. Clearly, the elementariness property also holds for ATL*, as it is included in SL[1G]. In particular, although it has not been explicitly stated, this property is crucial for most of the results achieved in literature about ATL* by means of automata (see [Schewe 2008], as an example). Moreover, we believe that our proof techniques are of independent interest and applicable to other logics as well.

Related works. Several works have focused on extensions of ATL* to incorporate more powerful strategic constructs. Among them, we recall *Alternating-Time* μCALCULUS (AμCALCULUS, for short) [Alur et al. 2002], *Game Logic* (GL, for short) [Alur et al. 2002], *Quantified Decision Modality* μCALCULUS (QDμ, for short) [Pinchinat 2007], *Coordination Logic* (CL, for short) [Finkbeiner and Schewe 2010], and some extensions of ATL* considered in [Brihaye et al. 2009]. AμCALCULUS and QDμ are intrinsically different from SL (as well as from CHP-SL and ATL*) as they are obtained by extending the propositional μ-calculus [Kozen 1983] with strategic modalities. CL is similar to QDμ but with LTL temporal operators instead of explicit fixpoint constructors. GL is strictly included in CHP-SL, in the case of two-player turn-based games, but it does not use any explicit treatment of strategies, neither it does the extensions of ATL* introduced in [Brihaye et al. 2009]. In particular, the latter work consider restrictions on the memory for strategy quantifiers. Thus, all above logics are different from SL, which we recall it aims to be a minimal but powerful logic to reason about strategic behavior in multi-agent systems. A very recent generalization of ATL*, which results to be expressive but a proper sublogic of SL, is also proposed in [Costa et al. 2010a]. In this logic, a quantification over strategies does not reset the strategies previously quantified but allows to maintain them in a particular context in order to be reused. This makes the logic much more expressive than ATL*. On the other hand, as it does not allow agents to share the same strategy, it is not comparable with the fragments we have considered in

this paper. Finally, we want to remark that our non-elementary hardness proof about the SL model-checking problem is inspired by and improves a proof proposed for their logic and communicated to us [Costa et al. 2010b] by the authors of [Costa et al. 2010a].

Note on [Mogavero et al. 2010a]. Preliminary results on SL appeared in [Mogavero et al. 2010a]. We presented there a 2EXPTIME algorithm for the model-checking problem. The described procedure applies only to the SL[1G] fragment, as model checking for full SL is non-elementary.

Outline. The remaining part of this work is structured as follows. In Section 2, we recall the semantic framework based on concurrent game structures and introduce syntax and semantics of SL. Then, in Section 3, we show the non-elementary lower bound for the model-checking problem. After this, in Section 4, we start the study of few syntactic and semantic SL fragments and introduce the concepts of dependence map and elementary satisfiability. Finally, in Section 5, we describe the model-checking automata-theoretic procedures for all SL fragments. Note that, in the accompanying Appendix A, we recall standard mathematical notation and some basic definitions that are used in the paper. However, for the sake of a simpler understanding of the technical part, we make a reminder, by means of footnotes, for each first use of a non trivial or immediate mathematical concept. The paper is self contained. All missing proofs in the main body of the work are reported in appendix.

2. STRATEGY LOGIC

In this section, we introduce *Strategy Logic*, an extension of the classic linear-time temporal logic LTL [Pnueli 1977] along with the concepts of strategy quantifications and agent binding. Our aim is to define a formalism that allows to express strategic plans over temporal goals in a way that separates the part related to the strategic reasoning from that concerning the tactical one. This distinctive feature is achieved by decoupling the instantiation of strategies, done through the quantifications, from their application by means of bindings. Our proposal, on the line marked by its precursor CHP-SL [Chatterjee et al. 2007; Chatterjee et al. 2010] and differently from classical temporal logics [Emerson 1990], turns in a logic that is not simply propositional but predicative, since we treat strategies as a first order concept via the use of agents and variables as explicit syntactic elements. This fact let us to write Boolean combinations and nesting of complex predicates, linked together by some common strategic choice, which may represent each one a different temporal goal. However, it is worth noting that the technical approach we follow here is quite different from that used for the definition of CHP-SL, which is based, on the syntactic side, on the CTL* formula framework [Emerson and Halpern 1986] and, on the semantic one, on the two-player turn-based game model [Perrin and Pin 2004].

The section is organized as follows. In Subsection 2.1, we recall the definition of concurrent game structure used to interpret Strategy Logic, whose syntax is introduced in Subsection 2.2. Then, in Subsection 2.3, we give, among the others, the notions of strategy and play, which are finally used, in Subsection 2.4, to define the semantics of the logic.

2.1. Underlying framework

As semantic framework for our logic language, we use a *graph-based model* for *multi-player games* named *concurrent game structure* [Alur et al. 2002]. Intuitively, this mathematical formalism provides a generalization of *Kripke structures* [Kripke 1963] and *labeled transition systems* [Keller 1976], modeling *multi-agent systems* viewed as games, in which players perform *concurrent actions* chosen strategically as a function on the history of the play.

Definition 2.1 (*Concurrent Game Structures*). A *concurrent game structure* (CGS, for short) is a tuple $\mathcal{G} \triangleq \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}, \mathrm{St}, \lambda, \tau, s_0 \rangle$, where AP and Ag are finite non-empty sets of *atomic propositions* and *agents*, Ac and St are enumerable non-empty sets of *actions* and *states*, $s_0 \in \mathrm{St}$ is a designated *initial state*, and $\lambda : \mathrm{St} \to 2^{\mathrm{AP}}$ is a *labeling function* that maps each state to the set of atomic propositions true in that state. Let $\mathrm{Dc} \triangleq \mathrm{Ac}^{\mathrm{Ag}}$ be the set of *decisions*, i.e., functions

from Ag to Ac representing the choices of an action for each agent. [1] Then, $\tau : \mathrm{St} \times \mathrm{Dc} \to \mathrm{St}$ is a *transition function* mapping a pair of a state and a decision to a state.

Observe that elements in St are not global states of the system, but states of the environment in which the agents operate. Thus, they can be viewed as states of the game, which do not include the local states of the agents. From a practical point of view, this means that all agents have perfect information on the whole game, since local states are not taken into account in the choice of actions [Fagin et al. 1995]. Observe also that, differently from other similar formalizations, each agent has the same set of possible executable actions, independently of the current state and of choices made by other agents. However, as already reported in literature [Pinchinat 2007], this simplifying choice does not result in a limitation of our semantics framework and allow us to give a simpler and clearer explanation of all formal definitions and techniques we work on.

From now on, apart from the examples and if not differently stated, all CGSs are defined on the same sets of atomic propositions AP and agents Ag, so, when we introduce a new structure in our reasonings, we do not make explicit their definition anymore. In addition, we use the italic letters p, a, c, and s, possibly with indexes, as meta-variables on, respectively, the atomic propositions $\mathsf{p}, \mathsf{q}, \dots$ in AP, the agents $\alpha, \beta, \gamma, \dots$ in Ag, the actions $0, 1, \dots$ in Ac, and the states $\mathsf{s}, \dots$ in St. Finally, we use the name of a CGS as a subscript to extract the components from its tuple-structure. Accordingly, if $\mathcal{G} = \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}, \mathrm{St}, \lambda, \tau, s_0 \rangle$, we have that $\mathrm{Ac}_{\mathcal{G}} = \mathrm{Ac}$, $\lambda_{\mathcal{G}} = \lambda$, $s_{0\mathcal{G}} = \mathsf{s}_0$, and so on. Furthermore, we use the same notational concept to make explicit to which CGS the set Dc of decisions is related to. Note that, we omit the subscripts if the structure can be unambiguously individuated from the context.

Now, to get attitude to the introduced semantic framework, let us describe two running examples of simple concurrent games. In particular, we start by modeling the *paper, rock, and scissor* game.

Example 2.2 (*Paper, Rock, and Scissor*). Consider the classic two-player concurrent game *paper, rock, and scissor* (PRS, for short) as represented in Figure 1, where a play continues until one of the participants catches the move of the other. Vertexes are states of the game and labels on edges represent decisions of agents or sets of them, where the symbol $*$ is used in place of every possible action. In this specific case, since there are only two agents, the pair of symbols

Fig. 1: The CGS $\mathcal{G}_{PRS}$.

$**$ indicates the whole set Dc of decisions. The agents "Alice" and "Bob" in $\mathrm{Ag} \triangleq \{\mathsf{A}, \mathsf{B}\}$ have as possible actions those in the set $\mathrm{Ac} \triangleq \{\mathsf{P}, \mathsf{R}, \mathsf{S}\}$, which stand for "paper", "rock", and "scissor", respectively. During the play, the game can stay in one of the three states in $\mathrm{St} \triangleq \{\mathsf{s}_i, \mathsf{s}_\mathsf{A}, \mathsf{s}_\mathsf{B}\}$, which represent, respectively, the waiting moment, named *idle*, and the two winner positions. The latter ones are labeled with one of the atomic propositions in $\mathrm{AP} \triangleq \{\mathsf{w}_\mathsf{A}, \mathsf{w}_\mathsf{B}\}$, in order to represent who is the winner. The catch of one action over another is described by the relation $C \triangleq \{(\mathsf{P}, \mathsf{R}), (\mathsf{R}, \mathsf{S}), (\mathsf{S}, \mathsf{P})\} \subseteq \mathrm{Ac} \times \mathrm{Ac}$. We can now define the CGS $\mathcal{G}_{PRS} \triangleq \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}, \mathrm{St}, \lambda, \tau, \mathsf{s}_i \rangle$ for the PRS game, with the labeling given by $\lambda(\mathsf{s}_i) \triangleq \emptyset$, $\lambda(\mathsf{s}_\mathsf{A}) \triangleq \{\mathsf{w}_\mathsf{A}\}$, and $\lambda(\mathsf{s}_\mathsf{B}) \triangleq \{\mathsf{w}_\mathsf{B}\}$ and the transition function set as follows, where $\mathrm{D}_\mathsf{A} \triangleq \{\mathsf{d} \in \mathrm{Dc}_{\mathcal{G}_{PRS}} : (\mathsf{d}(\mathsf{A}), \mathsf{d}(\mathsf{B})) \in C\}$ and $\mathrm{D}_\mathsf{B} \triangleq \{\mathsf{d} \in \mathrm{Dc}_{\mathcal{G}_{PRS}} : (\mathsf{d}(\mathsf{B}), \mathsf{d}(\mathsf{A})) \in C\}$ are the sets of winning decisions for the two agents: if $s = \mathsf{s}_i$ and $\mathsf{d} \in \mathrm{D}_\mathsf{A}$ then $\tau(s, \mathsf{d}) \triangleq \mathsf{s}_\mathsf{A}$, else if $s = \mathsf{s}_i$ and $\mathsf{d} \in \mathrm{D}_\mathsf{B}$ then $\tau(s, \mathsf{d}) \triangleq \mathsf{s}_\mathsf{B}$, otherwise $\tau(s, \mathsf{d}) \triangleq s$. Note that, when none of the two agents catches the action of the other, i.e., the used decision is in $\mathrm{D}_i \triangleq \mathrm{Dc}_{\mathcal{G}_{PRS}} \setminus (\mathrm{D}_\mathsf{A} \cup \mathrm{D}_\mathsf{B})$, the play remains in the idle state to allow another try, otherwise it is stuck in a winning position forever.
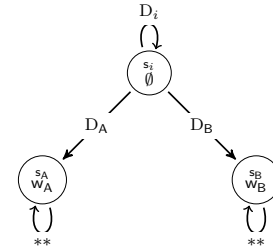
---

[1] In the following, we use both $\mathrm{X} \to \mathrm{Y}$ and Y^{X} to denote the set of functions from the domain X to the codomain Y.

We now describe a non-classic qualitative version of the well-known *prisoner's dilemma*.

Example 2.3 (*Prisoner's Dilemma*).   In the *prisoner's dilemma* (PD, for short), two accomplices are interrogated in separated rooms by the police, which offers them the same agreement. If one defects, i.e., testifies for the prosecution against the other, while the other co-operates, i.e., remains silent, the defector goes free and the silent ac-complice goes to jail. If both cooperate, they remain free, but will be surely interrogated in the next future waiting for a defection. On the other hand, if every one defects, both go to jail. It is ensured that no one will know about the choice made by the other. This

Fig. 2: The CGS $\mathcal{G}_{PD}$.

tricky situation can be modeled by the CGS $\mathcal{G}_{PD} \triangleq \langle \text{AP}, \text{Ag}, \text{Ac}, \text{St}, \lambda, \tau, \mathsf{s}_i \rangle$ depicted in Figure 2, where the agents "Accomplice-1" and "Accomplice-2" in $\text{Ag} \triangleq \{\mathsf{A}_1, \mathsf{A}_2\}$ can chose an action in $\text{Ac} \triangleq \{\mathsf{C}, \mathsf{D}\}$, which stand for "cooperation" and "defection", respectively. There are four states in $\text{St} \triangleq \{\mathsf{s}_i, \mathsf{s}_{\mathsf{A}_1}, \mathsf{s}_{\mathsf{A}_2}, \mathsf{s}_j\}$. In the idle state s_i the agents are waiting for the interrogation, while s_j rep-resents the jail for both of them. The remaining states $\mathsf{s}_{\mathsf{A}_1}$ and $\mathsf{s}_{\mathsf{A}_2}$ indicate, instead, the situations in which only one of the agents become definitely free. To characterize the different meaning of these states, we use the atomic propositions in $\text{AP} \triangleq \{\mathsf{f}_{\mathsf{A}_1}, \mathsf{f}_{\mathsf{A}_2}\}$, which denote who is "free", by defining the following labeling: $\lambda(\mathsf{s}_i) \triangleq \{\mathsf{f}_{\mathsf{A}_1}, \mathsf{f}_{\mathsf{A}_2}\}$, $\lambda(\mathsf{s}_{\mathsf{A}_1}) \triangleq \{\mathsf{f}_{\mathsf{A}_1}\}$, $\lambda(\mathsf{s}_{\mathsf{A}_2}) \triangleq \{\mathsf{f}_{\mathsf{A}_2}\}$, and $\lambda(\mathsf{s}_j) \triangleq \emptyset$. The transition function τ can be easily deduced by the figure.
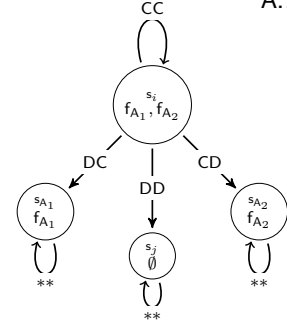
2.2. Syntax

Strategy Logic (SL, for short) syntactically extends LTL by means of two *strategy quantifiers*, the existential $\langle\langle x \rangle\rangle$ and the universal $[[x]]$, and *agent binding* (a, x), where a is an agent and x a variable. Intuitively, these new elements can be respectively read as *"there exists a strategy x"*, *"for all strategies x"*, and *"bind agent a to the strategy associated with the variable x"*. The formal syntax of SL follows.

Definition 2.4 (SL *Syntax*).   SL *formulas* are built inductively from the sets of atomic proposi-tions AP, variables Var, and agents Ag, by using the following grammar, where $p \in \text{AP}$, $x \in \text{Var}$, and $a \in \text{Ag}$:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathsf{X}\,\varphi \mid \varphi\,\mathsf{U}\,\varphi \mid \varphi\,\mathsf{R}\,\varphi \mid \langle\langle x \rangle\rangle\varphi \mid [[x]]\varphi \mid (a, x)\varphi.$$

SL denotes the infinite set of formulas generated by the above rules.

Observe that, by construction, LTL is a proper syntactic fragment of SL, i.e., $\text{LTL} \subset \text{SL}$. In order to abbreviate the writing of formulas, we use the boolean values true t and false f and the well-known temporal operators future $\mathsf{F}\,\varphi \triangleq \mathsf{t}\,\mathsf{U}\,\varphi$ and globally $\mathsf{G}\,\varphi \triangleq \mathsf{f}\,\mathsf{R}\,\varphi$. Moreover, we use the italic letters $x, y, z, \ldots$, possibly with indexes, as meta-variables on the variables $\mathsf{x}, \mathsf{y}, \mathsf{z}, \ldots$ in Var.

A first classic notation related to the SL syntax that we need to introduce is that of *subformula*, i.e., a syntactic expression that is part of an a priori given formula. By $\mathsf{sub} : \text{SL} \to 2^{\text{SL}}$ we formally denote the function returning the set of subformulas of an SL formula. For instance, consider $\varphi = \langle\langle \mathsf{x} \rangle\rangle(\alpha, \mathsf{x})(\mathsf{F}\,\mathsf{p})$. Then, it is immediate to see that $\mathsf{sub}(\varphi) = \{\varphi, (\alpha, \mathsf{x})(\mathsf{F}\,\mathsf{p}), (\mathsf{F}\,\mathsf{p}), \mathsf{p}, \mathsf{t}\}$.

Normally, predicative logics need the concepts of free and bound *placeholders* in order to for-mally define the meaning of their formulas. The placeholders are used to represent particular po-sitions in syntactic expressions that have to be highlighted, since they have a crucial role in the definition of the semantics. In first order logic, for instance, there is only one type of placeholders, which is represented by the variables. In SL, instead, we have both agents and variables as placehold-ers, as it can be noted by its syntax, in order to distinguish between the quantification of a strategy and its application by an agent. Consequently, we need a way to differentiate if an agent has an associated strategy via a variable and if a variable is quantified. To do this, we use the set of *free*

agents/variables as the subset of $\mathrm{Ag} \cup \mathrm{Var}$ containing *(i)* all agents for which there is no binding after the occurrence of a temporal operator and *(ii)* all variables for which there is a binding but no quantifications.

    Definition 2.5 (SL *Free Agents/Variables*).   The set of *free agents/variables* of an SL formula is given by the function free : $\mathrm{SL} \to 2^{\mathrm{Ag} \cup \mathrm{Var}}$ defined as follows:

  (i) free$(p) \triangleq \emptyset$, where $p \in \mathrm{AP}$;

  (ii) free$(\neg\varphi) \triangleq$ free(φ);

 (iii) free$(\varphi_1 \mathsf{Op}\ \varphi_2) \triangleq$ free$(\varphi_1) \cup$ free(φ_2), where $\mathsf{Op} \in \{\wedge, \vee\}$;

 (iv) free$(\mathsf{X}\ \varphi) \triangleq \mathrm{Ag} \cup$ free(φ);

  (v) free$(\varphi_1 \mathsf{Op}\ \varphi_2) \triangleq \mathrm{Ag} \cup$ free$(\varphi_1) \cup$ free(φ_2), where $\mathsf{Op} \in \{\mathsf{U}, \mathsf{R}\}$;

 (vi) free$(\mathsf{Qn}\ \varphi) \triangleq$ free$(\varphi) \setminus \{x\}$, where $\mathsf{Qn} \in \{\langle\!\langle x \rangle\!\rangle, [\![x]\!] : x \in \mathrm{Var}\}$;

 (vii) free$((a, x)\varphi) \triangleq$ free(φ), if $a \notin$ free(φ), where $a \in \mathrm{Ag}$ and $x \in \mathrm{Var}$;

(viii) free$((a, x)\varphi) \triangleq ($free$(\varphi) \setminus \{a\}) \cup \{x\}$, if $a \in$ free(φ), where $a \in \mathrm{Ag}$ and $x \in \mathrm{Var}$.

A formula φ without free agents (resp., variables), i.e., with free$(\varphi) \cap \mathrm{Ag} = \emptyset$ (resp., free$(\varphi) \cap \mathrm{Var} = \emptyset$), is named *agent-closed* (resp., *variable-closed*). If φ is both agent- and variable-closed, it is referred to as a *sentence*. The function snt : $\mathrm{SL} \to 2^{\mathrm{SL}}$ returns the set of *subsentences* snt$(\varphi) \triangleq \{\phi \in$ sub$(\varphi) :$ free$(\phi) = \emptyset\}$ for each SL formula φ.

Observe that, on one hand, free agents are introduced in Items iv and v and removed in Item viii. On the other hand, free variables are introduced in Item viii and removed in Item vi. As an example, let $\varphi = \langle\!\langle \mathsf{x} \rangle\!\rangle (\alpha, \mathsf{x})(\beta, \mathsf{y})(\mathsf{F}\ \mathsf{p})$ be a formula on the agents $\mathrm{Ag} = \{\alpha, \beta, \gamma\}$. Then, we have free$(\varphi) = \{\gamma, \mathsf{y}\}$, since γ is an agent without any binding after $\mathsf{F}\ \mathsf{p}$ and y has no quantification at all. Consider also the formulas $(\alpha, \mathsf{z})\varphi$ and $(\gamma, \mathsf{z})\varphi$, where the subformula φ is the same as above. Then, we have free$((\alpha, \mathsf{z})\varphi) =$ free(φ) and free$((\gamma, \mathsf{z})\varphi) = \{\mathsf{y}, \mathsf{z}\}$, since α is not free in φ but γ is, i.e., $\alpha \notin$ free(φ) and $\gamma \in$ free(φ). So, $(\gamma, \mathsf{z})\varphi$ is agent-closed while $(\alpha, \mathsf{z})\varphi$ is not.

    Similarly to the case of first order logic, another important concept that characterizes the syntax of SL is that of the *alternation number* of quantifiers, i.e., the maximum number of quantifier switches $\langle\!\langle \cdot \rangle\!\rangle [\![\cdot]\!]$, $[\![\cdot]\!]\langle\!\langle \cdot \rangle\!\rangle$, $\langle\!\langle \cdot \rangle\!\rangle \neg \langle\!\langle \cdot \rangle\!\rangle$, or $[\![\cdot]\!] \neg [\![\cdot]\!]$ that bind a variable in a subformula that is not a sentence. The constraint on the kind of subformulas that are considered here means that, when we evaluate the number of such switches, we consider each possible subsentence as an atomic proposition, hence, its quantifiers are not taken into account. Moreover, it is important to observe that vacuous quantifications, i.e., quantifications on variable that are not free in the immediate inner subformula, need to be not considered at all in the counting of quantifier switches. This value is crucial when we want to analyze the complexity of the decision problems of fragments of our logic, since higher alternation can usually mean higher complexity. By alt : $\mathrm{SL} \to \mathbb{N}$ we formally denote the function returning the alternation number of an SL formula. Furthermore, the fragment $\mathrm{SL}[k\text{-alt}] \triangleq \{\varphi \in \mathrm{SL} : \forall \varphi' \in$ sub(φ) . alt$(\varphi') \leq k\}$ of SL, for $k \in \mathbb{N}$, denotes the subset of formulas having all subformulas with alternation number bounded by k. For instance, consider the sentence $\varphi = [\![\mathsf{x}]\!]\langle\!\langle \mathsf{y} \rangle\!\rangle (\alpha, \mathsf{x})(\beta, \mathsf{y})(\mathsf{F}\ \varphi')$ with $\varphi' = [\![\mathsf{x}]\!]\langle\!\langle \mathsf{y} \rangle\!\rangle (\alpha, \mathsf{x})(\beta, \mathsf{y})(\mathsf{X}\ \mathsf{p})$, on the set of agents $\mathrm{Ag} = \{\alpha, \beta\}$. Then, the alternation number alt(φ) is 1 and not 3, as one can think at a first glance, since φ' is a sentence. Moreover, it holds that alt$(\varphi') = 1$. Hence, $\varphi \in \mathrm{SL}[1\text{-alt}]$. On the other hand, if we substitute φ' with $\varphi'' = [\![\mathsf{x}]\!](\alpha, \mathsf{x})(\mathsf{X}\ \mathsf{p})$, we have that alt$(\varphi) = 2$, since φ'' is not a sentence. Thus, it holds that $\varphi \notin \mathrm{SL}[1\text{-alt}]$ but $\varphi \in \mathrm{SL}[2\text{-alt}]$.

    At this point, in order to practice with the syntax of our logic by expressing game-theoretic concepts through formulas, we describe two examples of important properties that are possible to write in SL, but neither in ATL* [Alur et al. 2002] nor in CHP-SL. This is clarified later in the paper. The first concept we introduce is the well-known deterministic concurrent multi-player *Nash equilibrium* for Boolean valued payoffs.

Example 2.6 (*Nash Equilibrium*).   Consider the n agents $\alpha_1, \ldots, \alpha_n$ of a game, each of them having, respectively, a possibly different temporal goal described by one of the LTL formulas $\psi_1, \ldots, \psi_n$. Then, we can express the existence of a strategy profile $(x_1, \ldots, x_n)$ that is a *Nash equilibrium* (NE, for short) for $\alpha_1, \ldots, \alpha_n$ w.r.t. $\psi_1, \ldots, \psi_n$ by using the SL[1-alt] sentence $\varphi_{NE} \triangleq \langle\!\langle x_1 \rangle\!\rangle (\alpha_1, x_1) \cdots \langle\!\langle x_n \rangle\!\rangle (\alpha_n, x_n) \, \psi_{NE}$, where $\psi_{NE} \triangleq \bigwedge_{i=1}^{n} (\langle\!\langle y \rangle\!\rangle (\alpha_i, y)\psi_i) \to \psi_i$ is a variable-closed formula. Informally, this asserts that every agent α_i has x_i as one of the best strategy w.r.t. the goal ψ_i, once all the other strategies of the remaining agents α_j, with $j \neq i$, have been fixed to x_j. Note that here we are only considering equilibria under deterministic strategies.

As in physics, also in game theory an equilibrium is not always stable. Indeed, there are games like the PD of Example 2.3 on page 7 having Nash equilibria that are instable. One of the simplest concepts of stability that is possible to think is called *stability profile*.

Example 2.7 (*Stability Profile*).   Think about the same situation of the above example on NE. Then, a *stability profile* (SP, for short) is a strategy profile $(x_1, \ldots, x_n)$ for $\alpha_1, \ldots, \alpha_n$ w.r.t. $\psi_1, \ldots, \psi_n$ such that there is no agent α_i that can choose a different strategy from x_i without changing its own payoff and penalizing the payoff of another agent α_j, with $j \neq i$. To represent the existence of such a profile, we can use the SL[1-alt] sentence $\varphi_{SP} \triangleq \langle\!\langle x_1 \rangle\!\rangle (\alpha_1, x_1) \cdots \langle\!\langle x_n \rangle\!\rangle (\alpha_n, x_n) \psi_{SP}$, where $\psi_{SP} \triangleq \bigwedge_{i,j=1, i \neq j}^{n} \psi_j \to [\![y]\!]((\psi_i \leftrightarrow (\alpha_i, y)\psi_i) \to (\alpha_i, y)\psi_j)$. Informally, with the ψ_{SP} subformula, we assert that, if α_j is able to achieve his goal ψ_j, all strategies y of α_i that left unchanged the payoff related to ψ_i, also let α_j to maintain his achieved goal. At this point, it is very easy to ensure the existence of an NE that is also an SP, by using the SL[1-alt] sentence $\varphi_{SNE} \triangleq \langle\!\langle x_1 \rangle\!\rangle (\alpha_1, x_1) \cdots \langle\!\langle x_n \rangle\!\rangle (\alpha_n, x_n) \, \psi_{SP} \wedge \psi_{NE}$.

2.3. Basic concepts

Before continuing with the description of our logic, we have to introduce some basic concepts, regarding a generic CGS, that are at the base of the semantics formalization. Remind that a description of used mathematical notation is reported in Appendix A.

We start with the notions of *track* and *path*. Intuitively, tracks and paths of a CGS $\mathcal{G}$ are legal sequences of reachable states in $\mathcal{G}$ that can be respectively seen as partial and complete descriptions of possible outcomes of the game modeled by $\mathcal{G}$ itself.

Definition 2.8 (*Tracks and Paths*).   A *track* (resp., *path*) in a CGS $\mathcal{G}$ is a finite (resp., an infinite) sequence of states $\rho \in \mathrm{St}^*$ (resp., $\pi \in \mathrm{St}^\omega$) such that, for all $i \in [0, |\rho| - 1[$ (resp., $i \in \mathbb{N}$), there exists a decision $d \in \mathrm{Dc}$ such that $(\rho)_{i+1} = \tau((\rho)_i, d)$ (resp., $(\pi)_{i+1} = \tau((\pi)_i, d)$). [2] A track ρ is *non-trivial* if it has non-zero length, i.e., $|\rho| > 0$ that is $\rho \neq \varepsilon$. [3] The set $\mathrm{Trk} \subseteq \mathrm{St}^+$ (resp., $\mathrm{Pth} \subseteq \mathrm{St}^\omega$) contains all non-trivial tracks (resp., paths). Moreover, $\mathrm{Trk}(s) \triangleq \{\rho \in \mathrm{Trk} : \mathsf{fst}(\rho) = s\}$ (resp., $\mathrm{Pth}(s) \triangleq \{\pi \in \mathrm{Pth} : \mathsf{fst}(\pi) = s\}$) indicates the subsets of tracks (resp., paths) starting at a state $s \in \mathrm{St}$. [4]

For instance, consider the PRS game of Example 2.2 on page 6. Then, $\rho = s_i \cdot s_A \in \mathrm{St}^+$ and $\pi = s_i^\omega \in \mathrm{St}^\omega$ are, respectively, a track and a path in the CGS $\mathcal{G}_{PRS}$. Moreover, it holds that $\mathrm{Trk} = s_i^+ + s_i^* \cdot (s_A^+ + s_B^+)$ and $\mathrm{Pth} = s_i^\omega + s_i^* \cdot (s_A^\omega + s_B^\omega)$.

At this point, we can define the concept of *strategy*. Intuitively, a strategy is a scheme for an agent that contains all choices of actions as a function of the history of the current outcome. However, observe that here we do not set an a priori connection between a strategy and an agent, since the same strategy can be used by more than one agent at the same time.

---

[2] The notation $(w)_i \in \Sigma$ indicates the *element* of index $i \in [0, |w|[$ of a non-empty sequence $w \in \Sigma^\infty$.

[3] The Greek letter ε stands for the *empty sequence*.

[4] By $\mathsf{fst}(w) \triangleq (w)_0$ it is denoted the *first element* of a non-empty sequence $w \in \Sigma^\infty$.

Definition 2.9 (*Strategies*).   A *strategy* in a CGS $\mathcal{G}$ is a partial function $f : \mathrm{Trk} \rightharpoonup \mathrm{Ac}$ that maps each non-trivial track in its domain to an action. For a state $s \in \mathrm{St}$, a strategy f is said *s-total* if it is defined on all tracks starting in s, i.e., $\mathrm{dom}(f) = \mathrm{Trk}(s)$. The set $\mathrm{Str} \triangleq \mathrm{Trk} \rightharpoonup \mathrm{Ac}$ (resp., $\mathrm{Str}(s) \triangleq \mathrm{Trk}(s) \to \mathrm{Ac}$) contains all (resp., *s*-total) strategies.

An example of strategy in the CGS $\mathcal{G}_{PRS}$ is the function $f_1 \in \mathrm{Str}(s_i)$ that maps each track having length multiple of 3 to the action P, the tracks whose remainder of length modulo 3 is 1 to the action R, and the remaining tracks to the action S. A different strategy is given by the function $f_2 \in \mathrm{Str}(s_i)$ that returns the action P, if the tracks ends in s_A or s_B or if its length is neither a second nor a third power of a positive number, the action R, if the length is a square power, and the action S, otherwise.
    An important operation on strategies is that of *translation* along a given track, which is used to determine which part of a strategy has yet to be used in the game.

Definition 2.10 (*Strategy Translation*).   Let $f \in \mathrm{Str}$ be a strategy and $\rho \in \mathrm{dom}(f)$ a track in its domain. Then, $(f)_\rho \in \mathrm{Str}$ denotes the *translation* of f along ρ, i.e., the strategy with $\mathrm{dom}((f)_\rho) \triangleq \{\rho' \in \mathrm{Trk}(\mathrm{lst}(\rho)) : \rho \cdot \rho'_{\geq 1} \in \mathrm{dom}(f)\}$ such that $(f)_\rho(\rho') \triangleq f(\rho \cdot \rho'_{\geq 1})$, for all $\rho' \in \mathrm{dom}((f)_\rho)$. [5] [6]

Intuitively, the translation $(f)_\rho$ is the update of the strategy f, once the history of the game becomes ρ. It is important to observe that, if f is a $\mathrm{fst}(\rho)$-total strategy then $(f)_\rho$ is $\mathrm{lst}(\rho)$-total. For instance, consider the two tracks $\rho_1 = s_i{}^4 \in \mathrm{Trk}(s_i)$ and $\rho_2 = s_i{}^4 \cdot s_A{}^2 \in \mathrm{Trk}(s_i)$ in the CGS $\mathcal{G}_{PRS}$ and the strategy $f_1 \in \mathrm{Str}(s_i)$ previously described. Then, we have that $(f_1)_{\rho_1} = f_1$, while $(f_1)_{\rho_2} \in \mathrm{Str}(s_A)$ maps each track having length multiple of 3 to the action S, each track whose remainder of length modulo 3 is 1 to the action P, and the remaining tracks to the action R.
    We now introduce the notion of *assignment*. Intuitively, an assignment gives a valuation of variables with strategies, where the latter are used to determine the behavior of agents in the game. With more detail, as in the case of first order logic, we use this concept as a technical tool to quantify over strategies associated with variables, independently of agents to which they are related to. So, assignments are used precisely as a way to define a correspondence between variables and agents via strategies.

Definition 2.11 (*Assignments*).   An *assignment* in a CGS $\mathcal{G}$ is a partial function $\chi : \mathrm{Var} \cup \mathrm{Ag} \rightharpoonup \mathrm{Str}$ mapping variables and agents in its domain to a strategy. An assignment χ is *complete* if it is defined on all agents, i.e., $\mathrm{Ag} \subseteq \mathrm{dom}(\chi)$. For a state $s \in \mathrm{St}$, it is said that χ is *s-total* if all strategies $\chi(l)$ are *s*-total, for $l \in \mathrm{dom}(\chi)$. The set $\mathrm{Asg} \triangleq \mathrm{Var} \cup \mathrm{Ag} \rightharpoonup \mathrm{Str}$ (resp., $\mathrm{Asg}(s) \triangleq \mathrm{Var} \cup \mathrm{Ag} \rightharpoonup \mathrm{Str}(s)$) contains all (resp., *s*-total) assignments. Moreover, $\mathrm{Asg}(\mathrm{X}) \triangleq \mathrm{X} \to \mathrm{Str}$ (resp., $\mathrm{Asg}(\mathrm{X}, s) \triangleq \mathrm{X} \to \mathrm{Str}(s)$) indicates the subset of X-*defined* (resp., *s*-total) assignments, i.e., (resp., *s*-total) assignments defined on the set $\mathrm{X} \subseteq \mathrm{Var} \cup \mathrm{Ag}$.

As an example of assignment, let us consider the function $\chi_1 \in \mathrm{Asg}$ in the CGS $\mathcal{G}_{PRS}$, defined on the set $\{A, x\}$, whose values are f_1 on A and f_2 on x, where the strategies $f_1, f_2 \in \mathrm{Str}(s_i)$ are those described above. Another examples is given by the assignment $\chi_2 \in \mathrm{Asg}$, defined on the set $\{A, B\}$, such that $\chi_2(A) = \chi_1(x)$ and $\chi_2(B) = \chi_1(A)$. Note that both are s_i-total and the latter is also complete while the former is not.
    As in the case of strategies, it is useful to define the operation of *translation* along a given track for assignments too.

Definition 2.12 (*Assignment Translation*).   For a given state $s \in \mathrm{St}$, let $\chi \in \mathrm{Asg}(s)$ be an *s*-total assignment and $\rho \in \mathrm{Trk}(s)$ a track. Then, $(\chi)_\rho \in \mathrm{Asg}(\mathrm{lst}(\rho))$ denotes the *translation* of χ along ρ, i.e., the $\mathrm{lst}(\rho)$-total assignment, with $\mathrm{dom}((\chi)_\rho) \triangleq \mathrm{dom}(\chi)$, such that $(\chi)_\rho(l) \triangleq (\chi(l))_\rho$, for all $l \in \mathrm{dom}(\chi)$.

---

[5] By $\mathrm{lst}(w) \triangleq (w)_{|w|-1}$ it is denoted the *last element* of a finite non-empty sequence $w \in \Sigma^*$.
[6] The notation $(w)_{\geq i} \in \Sigma^\infty$ indicates the *suffix* from index $i \in [0, |w|]$ inwards of a non-empty sequence $w \in \Sigma^\infty$.

Intuitively, the translation $(\chi)_\rho$ is the simultaneous update of all strategies $\chi(l)$ defined by the assignment χ, once the history of the game becomes ρ.

Given an assignment χ, an agent or variable l, and a strategy f, it is important to define a notation to represent the *redefinition* of χ, i.e., a new assignment equal to the first on all elements of its domain but l, on which it assumes the value f.

Definition 2.13 (*Assignment Redefinition*). Let $\chi \in \mathrm{Asg}$ be an assignment, $\mathsf{f} \in \mathrm{Str}$ a strategy and $l \in \mathrm{Var} \cup \mathrm{Ag}$ either an agent or a variable. Then, $\chi[l \mapsto \mathsf{f}] \in \mathrm{Asg}$ denotes the new assignment defined on $\mathsf{dom}(\chi[l \mapsto \mathsf{f}]) \triangleq \mathsf{dom}(\chi) \cup \{l\}$ that returns f on l and is equal to χ on the remaining part of its domain, i.e., $\chi[l \mapsto \mathsf{f}](l) \triangleq \mathsf{f}$ and $\chi[l \mapsto \mathsf{f}](l') \triangleq \chi(l')$, for all $l' \in \mathsf{dom}(\chi) \setminus \{l\}$.

Intuitively, if we have to add or update a strategy that needs to be bound by an agent or variable, we can simply take the old assignment and redefine it by using the above notation. It is worth to observe that, if χ and f are s-total then $\chi[l \mapsto \mathsf{f}]$ is s-total too.

Now, we can introduce the concept of *play* in a game. Intuitively, a play is the unique outcome of the game determined by all agent strategies participating to it.

Definition 2.14 (*Plays*). A path $\pi \in \mathrm{Pth}(s)$ starting at a state $s \in \mathrm{St}$ is a *play* w.r.t. a complete s-total assignment $\chi \in \mathrm{Asg}(s)$ ((χ, s)-*play*, for short) if, for all $i \in \mathbb{N}$, it holds that $(\pi)_{i+1} = \tau((\pi)_i, \mathsf{d})$, where $\mathsf{d}(a) \triangleq \chi(a)((\pi)_{\leq i})$, for each $a \in \mathrm{Ag}$. [7] The partial function $\mathsf{play} : \mathrm{Asg} \times \mathrm{St} \rightharpoonup \mathrm{Pth}$, with $\mathsf{dom}(\mathsf{play}) \triangleq \{(\chi, s) : \mathrm{Ag} \subseteq \mathsf{dom}(\chi) \wedge \chi \in \mathrm{Asg}(s) \wedge s \in \mathrm{St}\}$, returns the (χ, s)-play $\mathsf{play}(\chi, s) \in \mathrm{Pth}(s)$, for all pairs (χ, s) in its domain.

As a last example, consider again the complete s_i-total assignment χ_2 previously described for the CGS $\mathcal{G}_{PRS}$, which returns the strategies f_2 and f_1 on the agents A and B, respectively. Then, we have that $\mathsf{play}(\chi_2, \mathsf{s}_i) = \mathsf{s}_i{}^3 \cdot \mathsf{s}_\mathsf{B}{}^\omega$. This means that the play is won by the agent B.

Finally, we give the definition of global translation of a complete assignment together with a related state, which is used to calculate, at a certain step of the play, what is the current state and its updated assignment.

Definition 2.15 (*Global Translation*). For a given state $s \in \mathrm{St}$ and a complete s-total assignment $\chi \in \mathrm{Asg}(s)$, the *i-th global translation* of (χ, s), with $i \in \mathbb{N}$, is the pair of a complete assignment and a state $(\chi, s)^i \triangleq ((\chi)_{(\pi)_{\leq i}}, (\pi)_i)$, where $\pi = \mathsf{play}(\chi, s)$.

In order to avoid any ambiguity of interpretation of the described notions, we may use the name of a CGS as a subscript of the sets and functions just introduced to clarify to which structure they are related to, as in the case of components in the tuple-structure of the CGS itself.

2.4. Semantics

As already reported at the beginning of this section, just like ATL^* and differently from CHP-SL, the semantics of SL is defined w.r.t. concurrent game structures. For a CGS $\mathcal{G}$, one of its states s, and an s-total assignment χ with $\mathsf{free}(\varphi) \subseteq \mathsf{dom}(\chi)$, we write $\mathcal{G}, \chi, s \models \varphi$ to indicate that the formula φ holds at s in $\mathcal{G}$ under χ. The semantics of SL formulas involving the atomic propositions, the Boolean connectives $\neg$, $\wedge$, and $\vee$, as well as the temporal operators X, U, and R is defined as usual in LTL. The novel part resides in the formalization of the meaning of strategy quantifications $\langle\!\langle x \rangle\!\rangle$ and $[\![x]\!]$ and agent binding (a, x).

Definition 2.16 (SL *Semantics*). Given a CGS $\mathcal{G}$, for all SL formulas φ, states $s \in \mathrm{St}$, and s-total assignments $\chi \in \mathrm{Asg}(s)$ with $\mathsf{free}(\varphi) \subseteq \mathsf{dom}(\chi)$, the modeling relation $\mathcal{G}, \chi, s \models \varphi$ is inductively defined as follows.

(1) $\mathcal{G}, \chi, s \models p$ if $p \in \lambda(s)$, with $p \in \mathrm{AP}$.
(2) For all formulas φ, φ_1, and φ_2, it holds that:

---

[7]The notation $(w)_{\leq i} \in \Sigma^*$ indicates the *prefix* up to index $i \in [0, |w|]$ of a non-empty sequence $w \in \Sigma^\infty$.

(a) $\mathcal{G}, \chi, s \models \neg\varphi$ if not $\mathcal{G}, \chi, s \models \varphi$, that is $\mathcal{G}, \chi, s \not\models \varphi$;

(b) $\mathcal{G}, \chi, s \models \varphi_1 \wedge \varphi_2$ if $\mathcal{G}, \chi, s \models \varphi_1$ and $\mathcal{G}, \chi, s \models \varphi_2$;

(c) $\mathcal{G}, \chi, s \models \varphi_1 \vee \varphi_2$ if $\mathcal{G}, \chi, s \models \varphi_1$ or $\mathcal{G}, \chi, s \models \varphi_2$.

(3) For a variable $x \in \text{Var}$ and a formula φ, it holds that:

(a) $\mathcal{G}, \chi, s \models \langle\!\langle x \rangle\!\rangle \varphi$ if there exists an s-total strategy $\mathsf{f} \in \text{Str}(s)$ such that $\mathcal{G}, \chi[x \mapsto \mathsf{f}], s \models \varphi$;

(b) $\mathcal{G}, \chi, s \models [\![x]\!]\varphi$ if for all s-total strategies $\mathsf{f} \in \text{Str}(s)$ it holds that $\mathcal{G}, \chi[x \mapsto \mathsf{f}], s \models \varphi$.

(4) For an agent $a \in \text{Ag}$, a variable $x \in \text{Var}$, and a formula φ, it holds that $\mathcal{G}, \chi, s \models (a, x)\varphi$ if $\mathcal{G}, \chi[a \mapsto \chi(x)], s \models \varphi$.

(5) Finally, if the assignment χ is also complete, for all formulas φ, φ_1, and φ_2, it holds that:

(a) $\mathcal{G}, \chi, s \models \mathsf{X}\,\varphi$ if $\mathcal{G}, (\chi, s)^1 \models \varphi$;

(b) $\mathcal{G}, \chi, s \models \varphi_1 \mathsf{U}\, \varphi_2$ if there is an index $i \in \mathbb{N}$ with $k \le i$ such that $\mathcal{G}, (\chi, s)^i \models \varphi_2$ and, for all indexes $j \in \mathbb{N}$ with $k \le j < i$, it holds that $\mathcal{G}, (\chi, s)^j \models \varphi_1$;

(c) $\mathcal{G}, \chi, s \models \varphi_1 \mathsf{R}\, \varphi_2$ if, for all indexes $i \in \mathbb{N}$ with $k \le i$, it holds that $\mathcal{G}, (\chi, s)^i \models \varphi_2$ or there is an index $j \in \mathbb{N}$ with $k \le j < i$ such that $\mathcal{G}, (\chi, s)^j \models \varphi_1$.

Intuitively, at Items 3a and 3b, respectively, we evaluate the existential $\langle\!\langle x \rangle\!\rangle$ and universal $[\![x]\!]$ quantifiers over strategies, by associating them to the variable x. Moreover, at Item 4, by means of an agent binding (a, x), we commit the agent a to a strategy associated with the variable x. It is evident that, due to Items 5a, 5b, and 5c, the LTL semantics is simply embedded into the SL one.

In order to complete the description of the semantics, we now give the classic notions of *model* and *satisfiability* of an SL sentence.

Definition 2.17 (SL *Satisfiability*). We say that a CGS $\mathcal{G}$ is a *model* of an SL sentence φ, in symbols $\mathcal{G} \models \varphi$, if $\mathcal{G}, \varnothing, s_0 \models \varphi$. [8] In general, we also say that $\mathcal{G}$ is a *model* for φ on $s \in \text{St}$, in symbols $\mathcal{G}, s \models \varphi$, if $\mathcal{G}, \varnothing, s \models \varphi$. An SL sentence φ is *satisfiable* if there is a model for it.

It remains to introduce the concepts of *implication* and *equivalence* between SL formulas, which are useful to describe transformations preserving the meaning of a specification.

Definition 2.18 (SL *Implication and Equivalence*). Given two SL formulas φ_1 and φ_2 with $\text{free}(\varphi_1) = \text{free}(\varphi_2)$, we say that φ_1 *implies* φ_2, in symbols $\varphi_1 \Rightarrow \varphi_2$, if, for all CGSs $\mathcal{G}$, states $s \in \text{St}$, and $\text{free}(\varphi_1)$-defined s-total assignments $\chi \in \text{Asg}(\text{free}(\varphi_1), s)$, it holds that if $\mathcal{G}, \chi, s \models \varphi_1$ then $\mathcal{G}, \chi, s \models \varphi_2$. Accordingly, we say that φ_1 is *equivalent* to φ_2, in symbols $\varphi_1 \equiv \varphi_2$, if both $\varphi_1 \Rightarrow \varphi_2$ and $\varphi_2 \Rightarrow \varphi_1$ hold.

In the rest of the paper, especially when we describe a decision procedure, we may consider formulas in *existential normal form* (*enf*, for short) and *positive normal form* (*pnf*, for short), i.e., formulas in which only existential quantifiers appear or in which the negation is applied only to atomic propositions. In fact, it is to this aim that we have considered in the syntax of SL both the Boolean connectives $\wedge$ and $\vee$, the temporal operators U, and R, and the strategy quantifiers $\langle\!\langle \cdot \rangle\!\rangle$ and $[\![\cdot]\!]$. Indeed, all formulas can be linearly translated in *pnf* by using De Morgan's laws together with the following equivalences, which directly follow from the semantics of the logic: $\neg\mathsf{X}\,\varphi \equiv \mathsf{X}\,\neg\varphi$, $\neg(\varphi_1 \mathsf{U}\, \varphi_2) \equiv (\neg\varphi_1)\mathsf{R}\,(\neg\varphi_2)$, $\neg\langle\!\langle x \rangle\!\rangle\varphi \equiv [\![x]\!]\neg\varphi$, and $\neg(a, x)\varphi \equiv (a, x)\neg\varphi$.

At this point, in order to better understand the meaning of our logic, we discuss two examples in which we describe the evaluation of the semantics of some formula w.r.t. the a priori given CGSs. We start by explaining how a strategy can be shared by different agents.

Example 2.19 (*Shared Variable*). Consider the SL[2-alt] sentence $\varphi = \langle\!\langle \mathsf{x} \rangle\!\rangle [\![\mathsf{y}]\!] \langle\!\langle \mathsf{z} \rangle\!\rangle ((\alpha, \mathsf{x})(\beta, \mathsf{y})(\mathsf{X}\,\mathsf{p}) \wedge (\alpha, \mathsf{y})(\beta, \mathsf{z})(\mathsf{X}\,\mathsf{q}))$. It is immediate to note that both agents α and β use the strategy associated with y to achieve simultaneously the LTL temporal goals $\mathsf{X}\,\mathsf{p}$

Fig. 3: The CGS $\mathcal{G}_{SV}$.

---

[8] The symbol $\varnothing$ stands for the empty function.

and X q. A model for φ is given by the CGS $\mathcal{G}_{SV} \triangleq \langle \{p, q\}, \{\alpha, \beta\},$
$\{0, 1\}, \{s_0, s_1, s_2, s_3\}, \lambda, \tau, s_0 \rangle$, where $\lambda(s_0) \triangleq \emptyset$, $\lambda(s_1) \triangleq \{p\}$, $\lambda(s_2) \triangleq \{p, q\}$, $\lambda(s_3) \triangleq \{q\}$,
$\tau(s_0, (0, 0)) \triangleq s_1$, $\tau(s_0, (0, 1)) \triangleq s_2$, $\tau(s_0, (1, 0)) \triangleq s_3$, and all the remaining transitions (with any
decision) go to s_0. In Figure 3 on the facing page, we report a graphical representation of the structure. Clearly, $\mathcal{G}_{SV} \models \varphi$ by letting, on s_0, the variables x to chose action 0 (the goal $(\alpha, x)(\beta, y)(X\ p)$
is satisfied for any choice of y, since we can move from s_0 to either s_1 or s_2, both labeled with p)
and z to choose action 1 when y has action 0 and, vice versa, 0 when y has 1 (in both cases, the goal
$(\alpha, y)(\beta, z)(X\ q)$ is satisfied, since one can move from s_0 to either s_2 or s_3, both labeled with q).

We now discuss an application of the concepts of Nash equilibrium and stability profile to both
the prisoner's dilemma and the paper, rock, and scissor game.

Example 2.20 (*Equilibrium Profiles*). Let us first to consider the CGS $\mathcal{G}_{PD}$ of the prisoner's
dilemma described in the Example 2.3 on page 7. Intuitively, each of the two accomplices A_1 and
A_2 want to avoid the prison. These goals can be, respectively, represented by the LTL formulas
$\psi_{A_1} \triangleq G\ f_{A_1}$ and $\psi_{A_2} \triangleq G\ f_{A_2}$. The existence of a Nash equilibrium in $\mathcal{G}_{PD}$ for the two accomplices
w.r.t. the above goals can be written as $\phi_{NE} \triangleq \langle\langle x_1 \rangle\rangle (A_1, x_1) \langle\langle x_2 \rangle\rangle (A_2, x_2)\ \psi_{NE}$, where $\psi_{NE} \triangleq$
$(((\langle\langle y \rangle\rangle (A_1, y) \psi_{A_1}) \rightarrow \psi_{A_1}) \wedge (((\langle\langle y \rangle\rangle (A_2, y) \psi_{A_2}) \rightarrow \psi_{A_2})$, which results to be an instantiation of the
general sentence φ_{NE} of Example 2.6 on page 9. In the same way, the existence of a stable Nash
equilibrium can be represented with the sentence $\phi_{SNE} \triangleq \langle\langle x_1 \rangle\rangle (A_1, x_1) \langle\langle x_2 \rangle\rangle (A_2, x_2)\ \psi_{NE} \wedge \psi_{SP}$,
where $\psi_{SP} \triangleq (\psi_1 \rightarrow [\![y]\!]((\psi_2 \leftrightarrow (A_2, y) \psi_2) \rightarrow (A_2, y) \psi_1)) \wedge (\psi_2 \rightarrow [\![y]\!]((\psi_1 \leftrightarrow (A_1, y) \psi_1) \rightarrow$
$(A_1, y) \psi_2))$, which is a particular case of the sentence φ_{SNE} of Example 2.7 on page 9. Now, it is
easy to see that $\mathcal{G}_{PD} \models \phi_{SNE}$ and, so, $\mathcal{G}_{PD} \models \phi_{NE}$. Indeed, an assignment $\chi \in \mathrm{Asg}_{\mathcal{G}_{PD}}(\mathrm{Ag}, s_i)$, for
which $\chi(A_1)(s_i) = \chi(A_2)(s_i) = D$, is a stable equilibrium profile, i.e., it is such that $\mathcal{G}_{PD}, \chi, s_i \models$
$\psi_{NE} \wedge \psi_{SP}$. This is due to the fact that, if an agent A_k, for $k \in \{1, 2\}$, choses another strategy
$f \in \mathrm{Str}_{\mathcal{G}_{PD}}(s_i)$, he is still unable to achieve his goal ψ_k, i.e., $\mathcal{G}_{PD}, \chi[A_k \mapsto f], s_i \not\models \psi_k$, so, he cannot
improve his payoff. Moreover, this equilibrium is stable, since the payoff of an agent cannot be made
worse by the changing of the strategy of the other agent. However, it is interesting to note that there
are instable equilibria too. One of these is represented by the assignment $\chi' \in \mathrm{Asg}_{\mathcal{G}_{PD}}(\mathrm{Ag}, s_i)$, for
which $\chi'(A_1)(s_i^j) = \chi'(A_2)(s_i^j) = C$, for all $j \in \mathbb{N}$. Indeed, we have that $\mathcal{G}_{PD}, \chi', s_i \models \psi_{NE}$, since
$\mathcal{G}_{PD}, \chi', s_i \models \psi_1$ and $\mathcal{G}_{PD}, \chi', s_i \models \psi_2$, but $\mathcal{G}_{PD}, \chi', s_i \not\models \psi_{SP}$. The latter property holds because,
if one of the agents A_k, for $k \in \{1, 2\}$, choses a different strategy $f' \in \mathrm{Str}_{\mathcal{G}_{PD}}(s_i)$ for which there
is a $j \in \mathbb{N}$ such that $f'(s_i^j) = D$, he cannot improve his payoff but makes surely worse the payoff of
the other agent, i.e., $\mathcal{G}_{PD}, \chi'[A_k \mapsto f'], s_i \models \psi_k$ but $\mathcal{G}_{PD}, \chi'[A_k \mapsto f'], s_i \not\models \psi_{3-k}$. Finally, consider
the CGS $\mathcal{G}_{PRS}$ of the paper, rock, and scissor game described in the Example 2.2 on page 6 together
with the associated formula for the Nash equilibrium $\phi_{NE} \triangleq \langle\langle x_1 \rangle\rangle (A, x_1) \langle\langle x_2 \rangle\rangle (B, x_2)\ \psi_{NE}$, where
$\psi_{NE} \triangleq (((\langle\langle y \rangle\rangle (A, y) \psi_A) \rightarrow \psi_A) \wedge (((\langle\langle y \rangle\rangle (B, y) \psi_B) \rightarrow \psi_B)$ with $\psi_A \triangleq F\ w_A$ and $\psi_B \triangleq F\ w_B$
representing the LTL temporal goals for Alice and Bob, respectively. Then, it is not hard to see that
$\mathcal{G}_{PRS} \not\models \phi_{NE}$, i.e., there are no Nash equilibria in this game, since there is necessarily an agent that
can improve his/her payoff by changing his/her strategy.

Finally, we want to remark that our semantics framework, based on concurrent game structures,
is enough expressive to describe turn-based features in the multi-agent case too. This is possible by
simply allowing the transition function to depend only on the choice of actions of an a priori given
agent for each state.

Definition 2.21 (*Turn-Based Game Structures*). A CGS $\mathcal{G}$ is *turn-based* if there exists a function $\eta : \mathrm{St} \rightarrow \mathrm{Ag}$, named *owner function*, such that, for all states $s \in \mathrm{St}$ and decisions $d_1, d_2 \in \mathrm{Dc}$,
it holds that if $d_1(\eta(s)) = d_2(\eta(s))$ then $\tau(s, d_1) = \tau(s, d_2)$.

Intuitively, a CGS is turn-based if it is possible to associate with each state an agent, i.e., the owner
of the state, which is responsible for the choice of the successor of that state. It is immediate to
observe that η introduces a partitioning of the set of states into $|\mathrm{rng}(\eta)|$ components, each one ruled

by a single agent. Moreover, observe that a CGS having just one agent is trivially turn-based, since this agent is the only possible owner of all states.

In the following, as one can expect, we also consider the case in which SL has its semantics defined on turn-based CGS only. In such an eventuality, we name the resulting semantic fragment *Turn-based Strategy Logic* (TB-SL, for short) and refer to the related satisfiability concept as *turn-based satisfiability*.

3. MODEL-CHECKING HARDNESS

In this section, we show the non-elementary lower bound for the model-checking problem of SL. Precisely, we prove that, for sentences having alternation number k, this problem is k-EXPSPACE-HARD. To this aim, in Subsection 3.1, we first recall syntax and semantics of QPTL [Sistla 1983]. Then, in Subsection 3.2, we give a reduction from the satisfiability problem for this logic to the model-checking problem for SL.

3.1. Quantified propositional temporal logic

Quantified Propositional Temporal Logic (QPTL, for short) syntactically extends the old-style temporal logic with the *future* F and *global* G operators by means of two *proposition quantifiers*, the existential $\exists q.$ and the universal $\forall q.$, where q is an atomic proposition. Intuitively, these elements can be respectively read as *"there exists an evaluation of q"* and *"for all evaluations of q"*. The formal syntax of QPTL follows.

Definition 3.1 (QPTL *Syntax*).   QPTL *formulas* are built inductively from the sets of atomic propositions AP, by using the following grammar, where $p \in$ AP:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathsf{X}\,\varphi \mid \mathsf{F}\,\varphi \mid \mathsf{G}\,\varphi \mid \exists p.\varphi \mid \forall p.\varphi.$$

QPTL denotes the infinite set of formulas generated by the above grammar.

Similarly to SL, we use the concepts of subformula, free atomic proposition, sentence, and alternation number, together with the QPTL syntactic fragment of bounded alternation QPTL[k-alt], with $k \in \mathbb{N}$.

In order to define the semantics of QPTL, we have first to introduce the concepts of truth evaluations used to interpret the meaning of atomic propositions at the passing of time.

Definition 3.2 (*Truth Evaluations*).   A *temporal truth evaluation* is a function tte $: \mathbb{N} \to \{\mathfrak{f}, \mathfrak{t}\}$ that maps each natural number to a Boolean value. Moreover, a *propositional truth evaluation* is a partial function pte $:$ AP $\rightharpoonup$ TTE mapping every atomic proposition in its domain to a temporal truth evaluation. The sets TTE $\triangleq \mathbb{N} \to \{\mathfrak{f}, \mathfrak{t}\}$ and PTE $\triangleq$ AP $\rightharpoonup$ TTE contain, respectively, all temporal and propositional truth evaluations.

At this point, we have the tool to define the interpretation of QPTL formulas. For a propositional truth evaluation pte with free(φ) $\subseteq$ dom(pte) and a number k, we write pte, $k \models \varphi$ to indicate that the formula φ holds at the k-th position of the pte.

Definition 3.3 (QPTL *Semantics*).   For all QPTL formulas φ, propositional truth evaluation pte $\in$ PTE with free(φ) $\subseteq$ dom(pte), and numbers $k \in \mathbb{N}$, the modeling relation pte, $k \models \varphi$ is inductively defined as follows.

(1) pte, $k \models p$ iff pte$(p)(k) = \mathfrak{t}$, with $p \in$ AP.
(2) For all formulas φ, φ_1, and φ_2, it holds that:
  (a) pte, $k \models \neg\varphi$ iff not pte, $k \models \varphi$, that is pte, $k \not\models \varphi$;
  (b) pte, $k \models \varphi_1 \wedge \varphi_2$ iff pte, $k \models \varphi_1$ and pte, $k \models \varphi_2$;
  (c) pte, $k \models \varphi_1 \vee \varphi_2$ iff pte, $k \models \varphi_1$ or pte, $k \models \varphi_2$;
  (d) pte, $k \models \mathsf{X}\,\varphi$ iff pte, $k + 1 \models \varphi$;
  (e) pte, $k \models \mathsf{F}\,\varphi$ iff there is an index $i \in \mathbb{N}$ with $k \leq i$ such that pte, $i \models \varphi$;

(f) pte, $k \models \mathsf{G}\,\varphi$ iff, for all indexes $i \in \mathbb{N}$ with $k \leq i$, it holds that pte, $i \models \varphi$.

(3) For an atomic proposition $q \in \mathrm{AP}$ and a formula φ, it holds that:

    (a) pte, $k \models \exists q.\varphi$ iff there exists a temporal truth evaluation tte $\in$ TTE such that pte$[q \mapsto$ tte$], k \models \varphi$;

    (b) pte, $k \models \forall q.\varphi$ iff for all temporal truth evaluations tte $\in$ TTE it holds that pte$[q \mapsto$ tte$], k \models \varphi$.

Obviously, a QPTL sentence φ is *satisfiable* if $\varnothing, 0 \models \varphi$. Observe that the described semantics is slightly different but completely equivalent to that proposed and used in [Sistla et al. 1987] to prove the non-elementary hardness result for the satisfiability problem.

3.2. Non-elementary lower-bound

We can show how the solution of QPTL satisfiability problem can be reduced to that of the model-checking problem for SL, over a turn-based constant size CGS with a unique atomic proposition.

In order to do this, we first prove the following auxiliary lemma, which actually represents the main step of the above mentioned reduction.
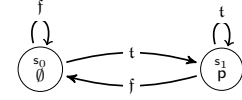

Fig. 4: The CGS $\mathcal{G}_{Rdc}$.

LEMMA 3.4 (QPTL REDUCTION). *There is a one-agent* CGS $\mathcal{G}_{Rdc}$ *such that, for each* QPTL[k-alt] *sentence* φ, *with* $k \in \mathbb{N}$, *there exists an* TB-SL[k-alt] *variable-closed formula* $\overline{\varphi}$ *such that* φ *is satisfiable iff* $\mathcal{G}_{Rdc}, \chi, \mathsf{s}_0 \models \overline{\varphi}$, *for all complete assignments* $\chi \in \mathrm{Asg}(\mathrm{Ag}, \mathsf{s}_0)$.

PROOF. Consider the one-agent CGS $\mathcal{G}_{Rdc} \triangleq \langle \{\mathsf{p}\}, \{\alpha\}, \{\mathsf{f}, \mathsf{t}\}, \{\mathsf{s}_0, \mathsf{s}_1\}, \lambda, \tau, \mathsf{s}_0 \rangle$ depicted in Figure 4, where the two actions are the Boolean values false and true and where the labeling and transition functions λ and τ are set as follows: $\lambda(\mathsf{s}_0) \triangleq \emptyset$, $\lambda(\mathsf{s}_1) \triangleq \{\mathsf{p}\}$, and $\tau(s, \mathsf{d}) = \mathsf{s}_0$ iff $\mathsf{d}(\alpha) = \mathsf{f}$, for all $s \in \mathrm{St}$ and $\mathsf{d} \in \mathrm{Dc}$. It is evident that $\mathcal{G}_{Rdc}$ is a turn-based CGS. Moreover, consider the transformation function $\overline{\cdot} : \mathrm{QPTL} \to \mathrm{SL}$ inductively defined as follows:

— $\overline{q} \triangleq (\alpha, \mathsf{x}_q)\mathsf{X}\,\mathsf{p}$, for $q \in \mathrm{AP}$;

— $\overline{\exists q.\varphi} \triangleq \langle\!\langle \mathsf{x}_q \rangle\!\rangle \overline{\varphi}$;

— $\overline{\forall q.\varphi} \triangleq [\![\mathsf{x}_q]\!] \overline{\varphi}$;

— $\overline{\mathsf{Op}\,\varphi} \triangleq \mathsf{Op}\,\overline{\varphi}$, where $\mathsf{Op} \in \{\neg, \mathsf{X}, \mathsf{F}, \mathsf{G}\}$;

— $\overline{\varphi_1 \mathsf{Op}\,\varphi_2} \triangleq \overline{\varphi_1}\mathsf{Op}\,\overline{\varphi_2}$, where $\mathsf{Op} \in \{\wedge, \vee\}$.

It is not hard to see that a QPTL formula φ is a sentence iff $\overline{\varphi}$ is variable-closed. Furthermore, we have that $\mathsf{alt}(\overline{\varphi}) = \mathsf{alt}(\varphi)$.

At this point, it remains to prove that, a QPTL sentence φ is satisfiable iff $\mathcal{G}_{Rdc}, \chi, \mathsf{s}_0 \models \overline{\varphi}$, for all total assignments $\chi \in \mathrm{Asg}(\{\alpha\}, \mathsf{s}_0)$. To do this by induction on the structure of φ, we actually show a stronger result asserting that, for all subformulas $\psi \in \mathrm{sub}(\varphi)$, propositional truth evaluations pte $\in$ PTE, and $i \in \mathbb{N}$, it holds that pte, $i \models \psi$ iff $\mathcal{G}_{Rdc}, (\chi, \mathsf{s}_0)^i \models \overline{\psi}$, for each total assignment $\chi \in \mathrm{Asg}(\{\alpha\} \cup \{\mathsf{x}_q \in \mathrm{Var} : q \in \mathrm{free}(\psi)\}, \mathsf{s}_0)$ such that $\chi(\mathsf{x}_q)((\pi)_{\leq n}) = \mathrm{pte}(q)(n)$, where $\pi \triangleq \mathsf{play}(\chi, \mathsf{s}_0)$, for all $q \in \mathrm{free}(\psi)$ and $n \in [i, \omega[$.

Here, we only show the base case of atomic propositions and the two inductive cases regarding the proposition quantifiers. The remaining cases of Boolean connectives and temporal operators are straightforward and left to the reader as a simple exercise.

— $\psi = q$.

By Item 1 of Definition 3.3 of QPTL semantics, we have that pte, $i \models q$ iff $\mathrm{pte}(q)(i) = \mathsf{t}$. Thus, due to the above constraint on the assignment, it follows that pte, $i \models q$ iff $\chi(\mathsf{x}_q)((\pi)_{\leq i}) = \mathsf{t}$. Now, by applying Items 4 and 5a of Definition 2.16 of SL semantics, we have that $\mathcal{G}_{Rdc}, (\chi, \mathsf{s}_0)^i \models (\alpha, \mathsf{x}_q)\mathsf{X}\,\mathsf{p}$ iff $\mathcal{G}_{Rdc}, (\chi'[\alpha \mapsto \chi'(\mathsf{x}_q)], s')^1 \models \mathsf{p}$, where $(\chi', s') = (\chi, \mathsf{s}_0)^i$. At this point, due to the particular structure of the CGS $\mathcal{G}_{Rdc}$, we have that $\mathcal{G}_{Rdc}, (\chi'[\alpha \mapsto \chi'(\mathsf{x}_q)], s')^1 \models \mathsf{p}$ iff $(\pi')_1 = \mathsf{s}_1$,

where $\pi' \triangleq \mathsf{play}(\chi'[\alpha \mapsto \chi'(\mathsf{x}_q)], s')$, which in turn is equivalent to $\chi'(\mathsf{x}_q)((\pi')_{\leq 0}) = \mathsf{t}$. So, $\mathcal{G}_{Rdc}, (\chi, \mathsf{s}_0)^i \models (\alpha, \mathsf{x}_q)\mathsf{X}\,\mathsf{p}$ iff $\chi'(\mathsf{x}_q)((\pi')_{\leq 0}) = \mathsf{t}$. Now, by observing that $(\pi')_{\leq 0} = (\pi)_i$ and using the above definition of χ', we obtain that $\chi'(\mathsf{x}_q)((\pi')_{\leq 0}) = \chi(\mathsf{x}_q)((\pi)_{\leq i})$. Hence, $\mathsf{pte}, i \models q$ iff $\mathsf{pte}(q)(i) = \chi(\mathsf{x}_q)((\pi)_{\leq i}) = \mathsf{t} = \chi'(\mathsf{x}_q)((\pi')_{\leq 0})$ iff $\mathcal{G}_{Rdc}, (\chi, \mathsf{s}_0)^i \models (\alpha, \mathsf{x}_q)\mathsf{X}\,\mathsf{p}$.

— $\psi = \exists q.\psi'$.

  [Only if]. If $\mathsf{pte}, i \models \exists q.\psi'$, by Item 3a of Definition 3.3, there exists a temporal truth evaluation $\mathsf{tte} \in \mathrm{TTE}$ such that $\mathsf{pte}[q \mapsto \mathsf{tte}], i \models \psi'$. Now, consider a strategy $\mathsf{f} \in \mathrm{Str}(\mathsf{s}_0)$ such that $\mathsf{f}((\pi)_{\leq n}) = \mathsf{tte}(n)$, for all $n \in [i, \omega[$. Then, it is evident that $\chi[\mathsf{x}_q \mapsto \mathsf{f}](\mathsf{x}_q)((\pi)_{\leq n}) = \mathsf{pte}[q \mapsto \mathsf{tte}](q')(n)$, for all $q' \in \mathsf{free}(\psi)$ and $n \in [i, \omega[$. So, by the inductive hypothesis, it follows that $\mathcal{G}_{Rdc}, (\chi[\mathsf{x}_q \mapsto \mathsf{f}], \mathsf{s}_0)^i \models \overline{\psi'}$. Thus, we have that $\mathcal{G}_{Rdc}, (\chi, \mathsf{s}_0)^i \models \langle\!\langle \mathsf{x}_q \rangle\!\rangle \overline{\psi'}$.

  [If]. If $\mathcal{G}_{Rdc}, (\chi, \mathsf{s}_0)^i \models \langle\!\langle \mathsf{x}_q \rangle\!\rangle \overline{\psi'}$, there exists a strategy $\mathsf{f} \in \mathrm{Str}(\mathsf{s}_0)$ such that $\mathcal{G}_{Rdc}, (\chi[\mathsf{x}_q \mapsto \mathsf{f}], \mathsf{s}_0)^i \models \overline{\psi'}$. Now, consider a temporal truth evaluation $\mathsf{tte} \in \mathrm{TTE}$ such that $\mathsf{tte}(n) = \mathsf{f}((\pi)_{\leq n})$, for all $n \in [i, \omega[$. Then, it is evident that $\chi[\mathsf{x}_q \mapsto \mathsf{f}](\mathsf{x}_{q'})((\pi)_{\leq n}) = \mathsf{pte}[q \mapsto \mathsf{tte}](q')(n)$, for all $q' \in \mathsf{free}(\psi)$ and $n \in [i, \omega[$. So, by the inductive hypothesis, it follows that $\mathsf{pte}[q \mapsto \mathsf{tte}], i \models \psi'$. Thus, by Item 3a of Definition 3.3, we have that $\mathsf{pte}, i \models \exists q.\psi'$.

— $\psi = \forall q.\psi'$.

  [Only if]. For each strategy $\mathsf{f} \in \mathrm{Str}(\mathsf{s}_0)$, consider a temporal truth evaluation $\mathsf{tte} \in \mathrm{TTE}$ such that $\mathsf{tte}(n) = \mathsf{f}((\pi)_{\leq n})$, for all $n \in [i, \omega[$. It is evident that $\chi[\mathsf{x}_q \mapsto \mathsf{f}](\mathsf{x}_{q'})((\pi)_{\leq n}) = \mathsf{pte}[q \mapsto \mathsf{tte}](q')(n)$, for all $q' \in \mathsf{free}(\psi)$ and $n \in [i, \omega[$. Now, since $\mathsf{pte}, i \models \forall q.\psi'$, by Item 3b of Definition 3.3, it follows that $\mathsf{pte}[q \mapsto \mathsf{tte}], i \models \psi'$. So, by the inductive hypothesis, for each strategy $\mathsf{f} \in \mathrm{Str}(\mathsf{s}_0)$, it holds that $\mathcal{G}_{Rdc}, (\chi[\mathsf{x}_q \mapsto \mathsf{f}], \mathsf{s}_0)^i \models \overline{\psi'}$. Thus, we have that $\mathcal{G}_{Rdc}, (\chi, \mathsf{s}_0)^i \models [\![\mathsf{x}_q]\!]\overline{\psi'}$.

  [If]. For each temporal truth evaluation $\mathsf{tte} \in \mathrm{TTE}$, consider a strategy $\mathsf{f} \in \mathrm{Str}(\mathsf{s}_0)$ such that $\mathsf{f}((\pi)_{\leq n}) = \mathsf{tte}(n)$, for all $n \in [i, \omega[$. It is evident that $\chi[\mathsf{x}_q \mapsto \mathsf{f}](\mathsf{x}_{q'})((\pi)_{\leq n}) = \mathsf{pte}[q \mapsto \mathsf{tte}](q')(n)$, for all $q' \in \mathsf{free}(\psi)$ and $n \in [i, \omega[$. Now, since $\mathcal{G}_{Rdc}, (\chi, \mathsf{s}_0)^i \models [\![\mathsf{x}_q]\!]\overline{\psi'}$, it follows that $\mathcal{G}_{Rdc}, (\chi[\mathsf{x}_q \mapsto \mathsf{f}], \mathsf{s}_0)^i \models \overline{\psi'}$. So, by the inductive hypothesis, for each temporal truth evaluation $\mathsf{tte} \in \mathrm{TTE}$, it holds that $\mathsf{pte}[q \mapsto \mathsf{tte}], i \models \psi'$. Thus, by Item 3b of Definition 3.3, we have that $\mathsf{pte}, i \models \forall q.\psi'$.

Thus, we are done with the proof. □

Now, we can show the full reduction that allows us to state the existence of a non-elementary lower-bound for the model-checking problem of TB-SL and, thus, of SL.

THEOREM 3.5 (TB-SL MODEL-CHECKING HARDNESS). *The model-checking problem for* TB-SL[k-alt] *is* k-EXPSPACE-HARD.

PROOF. Let φ be a QPTL[k-alt] sentence, $\overline{\varphi}$ the related TB-SL[k-alt] variable-closed formula, and $\mathcal{G}_{Rdc}$ the turn-based CGS of Lemma 3.4 of QPTL reduction. Then, by applying the previous mentioned lemma, it is easy to see that φ is satisfiable iff $\mathcal{G}_{Rdc} \models [\![\mathsf{x}]\!](\alpha, \mathsf{x})\overline{\varphi}$ iff $\mathcal{G}_{Rdc} \models \langle\!\langle \mathsf{x} \rangle\!\rangle (\alpha, \mathsf{x})\overline{\varphi}$. Thus, the satisfiability problem for QPTL can be reduced to the model-checking problem for TB-SL. Now, since the satisfiability problem for QPTL[k-alt] is k-EXPSPACE-HARD [Sistla et al. 1987], we have that the model-checking problem for TB-SL[k-alt] is k-EXPSPACE-HARD as well. □

The following corollary is an immediate consequence of the previous theorem.

COROLLARY 3.6 (SL MODEL-CHECKING HARDNESS). *The model-checking problem for* SL[k-alt] *is* k-EXPSPACE-HARD.

4. STRATEGY QUANTIFICATIONS

Since model checking for SL is non-elementary hard while the same problem for ATL* is only 2EXPTIME-COMPLETE, a question that naturally arises is whether there are proper fragments of SL of practical interest, still strictly subsuming ATL*, that reside in such a complexity gap. In this

section, we answer positively to this question and go even further. Precisely, we enlighten a fundamental property that, if satisfied, allows to retain a 2EXPTIME-COMPLETE model-checking problem. We refer to such a property as *elementariness*. To formally introduce this concept, we use the notion of *dependence map* as a machinery.

The remaining part of this section is organized as follows. In Subsection 4.1, we describe three syntactic fragments of SL, named SL[NG], SL[BG], and SL[1G], having the peculiarity to use strategy quantifications grouped in atomic blocks. Then, in Subsection 4.2, we define the notion of dependence map, which is used, in Subsection 4.3, to introduce the concept of elementariness. Finally, in Subsection 4.4, we prove a fundamental result, which is at the base of our elementary model-checking procedure for SL[1G].

4.1. Syntactic fragments

In order to formalize the syntactic fragments of SL we want to investigate, we first need to define the concepts of *quantification* and *binding prefixes*.

Definition 4.1 (*Prefixes*). A *quantification prefix* over a set $V \subseteq Var$ of variables is a finite word $\wp \in \{\langle\!\langle x \rangle\!\rangle, [\![x]\!] : x \in V\}^{|V|}$ of length $|V|$ such that each variable $x \in V$ occurs just once in $\wp$, i.e., there is exactly one index $i \in [0, |V|[$ such that $(\wp)_i \in \{\langle\!\langle x \rangle\!\rangle, [\![x]\!]\}$. A *binding prefix* over a set of variables $V \subseteq Var$ is a finite word $\flat \in \{(a,x) : a \in Ag \wedge x \in V\}^{|Ag|}$ of length $|Ag|$ such that each agent $a \in Ag$ occurs just once in $\flat$, i.e., there is exactly one index $i \in [0, |Ag|[$ for which $(\flat)_i \in \{(a,x) : x \in V\}$. Finally, $Qnt(V) \subseteq \{\langle\!\langle x \rangle\!\rangle, [\![x]\!] : x \in V\}^{|V|}$ and $Bnd(V) \subseteq \{(a,x) : a \in Ag \wedge x \in V\}^{|Ag|}$ denote, respectively, the sets of all quantification and binding prefixes over variables in V.

We now have all tools to define the syntactic fragments we want to analyze, which we name, respectively, *Nested-Goal*, *Boolean-Goal*, and *One-Goal Strategy Logic* (SL[NG], SL[BG], and SL[1G], for short). For *goal* we mean an SL agent-closed formula of the kind $\flat\varphi$, with $Ag \subseteq free(\varphi)$, being $\flat \in Bnd(Var)$ a binding prefix. The idea behind SL[NG] is that, when there is a quantification over a variable used in a goal, we are forced to quantify over all free variables of the inner subformula containing the goal itself, by using a quantification prefix. In this way, the subformula is build only by nesting and Boolean combinations of goals. In addition, with SL[BG] we avoid nested goals sharing the variables of a same quantification prefix, but allow their Boolean combinations. Finally, SL[1G] forces the use of a different quantification prefix for each single goal in the formula. The formal syntax of SL[NG], SL[BG], and SL[1G] follows.

Definition 4.2 (SL[NG]*,* SL[BG]*, and* SL[1G] *Syntax*). SL[NG] formulas are built inductively from the sets of atomic propositions AP, quantification prefixes $Qnt(V)$ for any $V \subseteq Var$, and binding prefixes $Bnd(Var)$, by using the following grammar, with $p \in AP$, $\wp \in \cup_{V \subseteq Var} Qnt(V)$, and $\flat \in Bnd(Var)$:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathsf{X}\,\varphi \mid \varphi\,\mathsf{U}\,\varphi \mid \varphi\,\mathsf{R}\,\varphi \mid \wp\varphi \mid \flat\varphi,$$

where in the formation rule $\wp\varphi$ it is ensured that φ is agent-closed and $\wp \in Qnt(free(\varphi))$.
In addition, SL[BG] formulas are determined by splitting the above syntactic class in two different parts, of which the second is dedicated to build the Boolean combinations of goals avoiding their nesting:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathsf{X}\,\varphi \mid \varphi\,\mathsf{U}\,\varphi \mid \varphi\,\mathsf{R}\,\varphi \mid \wp\psi,$$
$$\psi ::= \flat\varphi \mid \neg\psi \mid \psi \wedge \psi \mid \psi \vee \psi,$$

where in the formation rule $\wp\psi$ it is ensured that $\wp \in Qnt(free(\psi))$.
Finally, the simpler SL[1G] formulas are obtained by forcing each goal to be coupled with a quantification prefix:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathsf{X}\,\varphi \mid \varphi\,\mathsf{U}\,\varphi \mid \varphi\,\mathsf{R}\,\varphi \mid \wp\flat\varphi,$$

where in the formation rule $\wp\flat\varphi$ it is ensured that $\wp \in \mathrm{Qnt}(\mathsf{free}(\flat\varphi))$.
$\mathrm{SL} \supset \mathrm{SL[NG]} \supset \mathrm{SL[BG]} \supset \mathrm{SL[1G]}$ denotes the syntactic chain of infinite sets of formulas generated by the respective grammars with the associated constraints on free variables of goals.

Intuitively, in $\mathrm{SL[NG]}$, $\mathrm{SL[BG]}$, and $\mathrm{SL[1G]}$, we force the writing of formulas to use atomic blocks of quantifications and bindings, where the related free variables are strictly coupled with those that are effectively quantified in the prefix just before the binding. In a nutshell, we can only write formulas by using sentences of the form $\wp\psi$ belonging to a kind of *prenex normal form* in which the quantifications contained into the *matrix* ψ only belong to the prefixes $\wp'$ of some inner subsentence $\wp'\psi' \in \mathsf{snt}(\wp\psi)$.

An $\mathrm{SL[NG]}$ sentence ϕ is *principal* if it is of the form $\phi = \wp\psi$, where ψ is agent-closed and $\wp \in \mathrm{Qnt}(\mathsf{free}(\psi))$. By $\mathsf{psnt}(\varphi) \subseteq \mathsf{snt}(\varphi)$ we denote the set of all principal subsentences of the formula φ.

We now introduce other two general restrictions in which the numbers $|\mathrm{Ag}|$ of agents and $|\mathrm{Var}|$ of variables that are used to write a formula are fixed to the a priori values $n, m \in [1, \omega[$, respectively. Moreover, we can also forbid the sharing of variables, i.e., each variable is binded to one agent only, so, we cannot force two agents to use the same strategy. We name these three fragments $\mathrm{SL}[n\text{-ag}]$, $\mathrm{SL}[m\text{-var}]$, and $\mathrm{SL[fvs]}$, respectively. Note that, in the one agent fragment, the restriction on the sharing of variables between agents, naturally, does not act, i.e., $\mathrm{SL[1\text{-ag}, fvs]} = \mathrm{SL[1\text{-ag}]}$.

To start to practice with the above fragments, consider again the sentence φ of Example 2.19 on page 12. It is easy to see that it actually belongs to $\mathrm{SL[BG, 2\text{-ag}, 3\text{-var}, 2\text{-alt}]}$, and so, to $\mathrm{SL[NG]}$, but not to $\mathrm{SL[1G]}$, since it is of the form $\wp(\flat_1 \mathsf{X}\, \mathsf{p} \wedge \flat_2 \mathsf{X}\, \mathsf{q})$, where the quantification prefix is $\wp = \langle\langle \mathsf{x} \rangle\rangle [[\mathsf{y}]] \langle\langle \mathsf{z} \rangle\rangle$ and the binding prefixes of the two goals are $\flat_1 = (\alpha, \mathsf{x})(\beta, \mathsf{y})$ and $\flat_2 = (\alpha, \mathsf{y})(\beta, \mathsf{z})$.

Along the paper, sometimes we assert that a given formula φ belongs to an SL syntactic fragment also if its syntax does not precisely correspond to what is described by the relative grammar. We do this in order to make easier the reading and interpretation of the formula φ itself and only in the case that it is simple to translate it into an equivalent formula that effectively belongs to the intended logic, by means of a simple generalization of classic rules used to put a formula of first order logic in the prenex normal form. For example, consider the sentence φ_{NE} of Example 2.6 on page 9 representing the existence of a Nash equilibrium. This formula is considered to belong to $\mathrm{SL[BG}, n\text{-ag}, 2n\text{-var, fvs, 1\text{-alt}]}$, since it can be easily translated in the form $\phi_{NE} = \wp \bigwedge_{i=1}^{n} \flat_i \psi_i \to \flat\psi_i$, where $\wp = \langle\langle \mathsf{x}_1 \rangle\rangle \cdots \langle\langle \mathsf{x}_n \rangle\rangle [[\mathsf{y}_1]] \cdots [[\mathsf{y}_n]]$, $\flat = (\alpha_1, \mathsf{x}_1) \cdots (\alpha_n, \mathsf{x}_n)$, $\flat_i = (\alpha_1, \mathsf{x}_1) \cdots (\alpha_{i-1}, \mathsf{x}_{i-1})(\alpha_i, \mathsf{y}_i)(\alpha_{i+1}, \mathsf{x}_{i+1}) \cdots (\alpha_n, \mathsf{x}_n)$, and $\mathsf{free}(\psi_i) = \mathrm{Ag}$. As another example, consider the sentence φ_{SP} of Example 2.7 on page 9 representing the existence of a stability profile. Also this formula is considered to belong to $\mathrm{SL[BG}, n\text{-ag}, 2n\text{-var, fvs, 1\text{-alt}]}$, since it is equivalent to $\phi_{SP} = \wp \bigwedge_{i,j=1, i \neq j}^{n} \flat\psi_j \to ((\flat\psi_i \leftrightarrow \flat_i\psi_i) \to \flat_i\psi_j)$. Note that both ϕ_{NE} and ϕ_{SP} are principal sentences.

Now, it is interesting to observe that CTL^* and ATL^* are exactly equivalent to $\mathrm{SL[1G, fvs, 0\text{-alt}]}$ and $\mathrm{SL[1G, fvs, 1\text{-alt}]}$, respectively. Moreover, GL [Alur et al. 2002] is the very simple fragment of $\mathrm{SL[BG, fvs, 1\text{-alt}]}$ that forces all goals in a formula to have a common part containing all variables quantified before the unique possible alternation of the quantification prefix. Finally, we have that $\mathrm{CHP\text{-}SL}$ is the $\mathrm{TB\text{-}SL[BG, 2\text{-ag}, fvs]}$ fragment.

Remark 4.3 ($\mathrm{TB\text{-}SL[NG]}$ *Model-Checking Hardness*). It is well-known that the non-elementary hardness result for the satisfiability problem of QPTL [Sistla et al. 1987] already holds for formulas in prenex normal form. Now, it is not hard to see that the transformation described in Lemma 3.4 of QPTL reduction puts $\mathrm{QPTL}[k\text{-alt}]$ sentences φ in prenex normal form into $\mathrm{TB\text{-}SL[NG, 1\text{-ag}, k\text{-alt}]}$ variable-closed formulas $\overline{\varphi} = \wp\psi$. Moreover, the derived $\mathrm{TB\text{-}SL[1\text{-ag}, k\text{-alt}]}$ sentence $\langle\langle \mathsf{x} \rangle\rangle(\alpha, \mathsf{x})\wp\psi$ used in Theorem 3.5 of $\mathrm{TB\text{-}SL}$ model-checking hardness is equivalent to the $\mathrm{TB\text{-}SL[NG, 1\text{-ag}, k\text{-alt}]}$ principal sentence $\langle\langle \mathsf{x} \rangle\rangle\wp(\alpha, \mathsf{x})\psi$, since x is not used in the quantification prefix $\wp$. Thus, the hardness result for the model-checking problem holds for $\mathrm{TB\text{-}SL[NG, 1\text{-ag}, k\text{-alt}]}$ and, consequently,

for $\textsc{Sl}[\textsc{ng}, 1\text{-ag}, k\text{-alt}]$ as well. However, it is important to observe that, unfortunately, it is not know if such an hardness result holds for $\textsc{Tb-Sl}[\textsc{bg}]$ or $\textsc{Sl}[\textsc{bg}]$ and, in particular, for $\textsc{Chp-Sl}$. We leave this problem open here.

(a) CGS $\mathcal{G}_1$.   (b) CGS $\mathcal{G}_2$.

Fig. 5: Alternation-2 non-equivalent CGSs.

At this point, we prove that $\textsc{Atl}^*$ is strictly less expressive than $\textsc{Sl}[1\textsc{g}]$ and, consequently, than $\textsc{Sl}[\textsc{bg}]$ and $\textsc{Sl}[\textsc{ng}]$. To do this, we show the existence of two structures that result to be equivalent only w.r.t. sentences having alternation number bounded by 1. It can be interesting to note that, we use an ad-hoc technique based on a brute-force check to verify that all $\textsc{Atl}^*$ formulas cannot distinguish between the two structures. A possible future line of research is to study variants of the Ehrenfeucht-Fraïssé game [Ebbinghaus and Flum 1995; Hodges 1993] for $\textsc{Sl}$, which allow to determine whether two structures are or not equivalent w.r.t. a particular $\textsc{Sl}$ fragment.

THEOREM 4.4 ($\textsc{Sl}[1\textsc{g}]$ VS $\textsc{Atl}^*$ EXPRESSIVENESS). *There exists an* $\textsc{Sl}[1\textsc{g}, 3\text{-ag}, \text{fvs}, 2\text{-alt}]$ *sentence having no* $\textsc{Atl}^*$ *equivalent.*

PROOF. Consider the two CGSs $\mathcal{G}_1 \triangleq \langle \{\mathsf{p}\}, \{\alpha, \beta, \gamma\}, \{0,1\}, \{\mathsf{s}_0, \mathsf{s}_1, \mathsf{s}_2\}, \lambda, \tau_1, \mathsf{s}_0\rangle$ and $\mathcal{G}_2 \triangleq \langle \{\mathsf{p}\}, \{\alpha, \beta, \gamma\}, \{0,1,2\}, \{\mathsf{s}_0, \mathsf{s}_1, \mathsf{s}_2\}, \lambda, \tau_2, \mathsf{s}_0\rangle$ depicted in Figure 5, where $\lambda(\mathsf{s}_0) = \lambda(\mathsf{s}_2) \triangleq \emptyset$, $\lambda(\mathsf{s}_1) \triangleq \{\mathsf{p}\}$, $\mathrm{D}_1 \triangleq \{00*, 11*\}$, and $\mathrm{D}_2 \triangleq \{00*, 11*, 12*, 200, 202, 211\}$. Moreover, consider the $\textsc{Sl}[1\textsc{g}, 3\text{-ag}, \text{fvs}, 2\text{-alt}]$ sentence $\varphi^* \triangleq \wp^* \flat^* \mathsf{X}\, \mathsf{p}$, where $\wp^* \triangleq [\![\mathsf{x}]\!]\langle\!\langle\mathsf{y}\rangle\!\rangle[\![z]\!]$ and $\flat^* \triangleq (\alpha, \mathsf{x})(\beta, \mathsf{y})(\gamma, \mathsf{z})$. Then, it is easy to see that $\mathcal{G}_1 \models \varphi^*$ but $\mathcal{G}_2 \not\models \varphi^*$. Indeed, $\mathcal{G}_1, \chi_1, \mathsf{s}_0 \models \flat^* \mathsf{X}\, \mathsf{p}$, for all $\chi_1 \in \mathrm{Asg}_{\mathcal{G}_1}(\{\mathsf{x}, \mathsf{y}, \mathsf{z}\}, \mathsf{s}_0)$ such that $\chi_1(\mathsf{y})(\mathsf{s}_0) = \chi_1(\mathsf{x})(\mathsf{s}_0)$, and $\mathcal{G}_2, \chi_2, \mathsf{s}_0 \models \flat^* \mathsf{X}\, \neg\mathsf{p}$, for all $\chi_2 \in \mathrm{Asg}_{\mathcal{G}_2}(\{\mathsf{x}, \mathsf{y}, \mathsf{z}\}, \mathsf{s}_0)$ such that $\chi_2(\mathsf{x})(\mathsf{s}_0) = 2$ and $\chi_2(\mathsf{z})(\mathsf{s}_0) = (\chi_2(\mathsf{y})(\mathsf{s}_0) + 1) \bmod 3$.

Now, due to the particular structure of the CGSs $\mathcal{G}_i$ under exam, with $i \in \{1, 2\}$, for each path $\pi \in \mathrm{Pth}_{\mathcal{G}_i}(\mathsf{s}_0)$, we have that either $\lambda((\pi)_j) = \{\mathsf{p}\}$ or $\lambda((\pi)_j) = \emptyset$, for all $j \in [1, \omega[$, i.e., apart from the initial state, the path is completely labeled either with $\{\mathsf{p}\}$ or with $\emptyset$. Thus, it is easy to see that, for each $\textsc{Atl}^*$ formula $\wp\flat\psi$, there is a literal $l_\psi \in \{\mathsf{p}, \neg\mathsf{p}\}$ such that $\mathcal{G}_i \models \wp\flat\psi$ iff $\mathcal{G}_i \models \wp\flat\mathsf{X}l_\psi$, for all $i \in \{1, 2\}$. W.l.o.g., we can suppose that $\flat = \flat^*$, since we are always able to uniformly rename the variables of the quantification and binding prefixes without changing the meaning of the sentence.

At this point, it is easy to see that there exists an index $k \in \{1, 2, 3\}$ for which it holds that either $\wp_k\flat^*\mathsf{X}l_\psi \Rightarrow \wp\flat^*\mathsf{X}l_\psi$ or $\wp\flat^*\mathsf{X}l_\psi \Rightarrow \overline{\wp_k}\flat^*\mathsf{X}l_\psi$, where $\wp_1 \triangleq [\![\mathsf{x}]\!][\![z]\!]\langle\!\langle\mathsf{y}\rangle\!\rangle$, $\wp_2 \triangleq \langle\!\langle\mathsf{x}\rangle\!\rangle\langle\!\langle\mathsf{y}\rangle\!\rangle[\![z]\!]$, and $\wp_3 \triangleq [\![\mathsf{y}]\!][\![z]\!]\langle\!\langle\mathsf{x}\rangle\!\rangle$. Thus, to prove that every $\textsc{Atl}^*$ formula cannot distinguish between $\mathcal{G}_1$ and $\mathcal{G}_2$, we can simply show that the sentences $\wp_k\flat^*\mathsf{X}l$, with $k \in \{1, 2, 3\}$ and $l \in \{\mathsf{p}, \neg\mathsf{p}\}$, do the same. In fact, it holds that $\mathcal{G}_i \models \wp_k\flat^*\mathsf{X}l$, for all $i \in \{1, 2\}$, $k \in \{1, 2, 3\}$, and $l \in \{\mathsf{p}, \neg\mathsf{p}\}$. Hence, the thesis holds. The check of the latter fact is trivial and left to the reader as an exercise. □

4.2. Dependence Maps

We now introduce the concept of dependence map of a quantification and show how any quantification prefix contained into an $\textsc{Sl}$ formula can be represented by an adequate choice of a dependence map over strategies. The main idea here is inspired by what Skolem proposed for the first order

logic in order to eliminate each existential quantification over variables, by substituting them with second order existential quantifications over functions, whose choice is uniform w.r.t. the universal variables.

First, we introduce some notation regarding quantification prefixes. Let $\wp \in \mathrm{Qnt}(\mathrm{V})$ be a quantification prefix over a set $\mathrm{V}(\wp) \triangleq \mathrm{V} \subseteq \mathrm{Var}$ of variables. By $\langle\langle\wp\rangle\rangle \triangleq \{x \in \mathrm{V}(\wp) : \exists i \in [0, |\wp|[\, . \, (\wp)_i = \langle\langle x\rangle\rangle\}$ and $[[\wp]] \triangleq \mathrm{V}(\wp) \setminus \langle\langle\wp\rangle\rangle$ we denote, respectively, the sets of *existential* and *universal variables* quantified in $\wp$. For two variables $x, y \in \mathrm{V}(\wp)$, we say that x *precedes* y in $\wp$, in symbols $x <_\wp y$, if x occurs before y in $\wp$, i.e., there are two indexes $i, j \in [0, |\wp|[$, with $i < j$, such that $(\wp)_i \in \{\langle\langle x\rangle\rangle, [[x]]\}$ and $(\wp)_j \in \{\langle\langle y\rangle\rangle, [[y]]\}$. Moreover, we say that y is *functional dependent* on x, in symbols $x \rightsquigarrow_\wp y$, if $x \in [[\wp]]$, $y \in \langle\langle\wp\rangle\rangle$, and $x <_\wp y$, i.e., y is existentially quantified after that x is universally quantified, so, there may be a dependence between a value chosen by x and that chosen by y. This definition induces the set $\mathrm{Dep}(\wp) \triangleq \{(x, y) \in \mathrm{V}(\wp) \times \mathrm{V}(\wp) : x \rightsquigarrow_\wp y\}$ of *dependence pairs* and its derived version $\mathrm{Dep}(\wp, y) \triangleq \{x \in \mathrm{V}(\wp) : x \rightsquigarrow_\wp y\}$ containing all variables from which y depends. Finally, we use $\overline{\wp} \in \mathrm{Qnt}(\mathrm{V}(\wp))$ to indicate the quantification derived from $\wp$ by *dualizing* each quantifier contained in it, i.e., for all indexes $i \in [0, |\wp|[$, it holds that $(\overline{\wp})_i = \langle\langle x\rangle\rangle$ iff $(\wp)_i = [[x]]$, with $x \in \mathrm{V}(\wp)$. It is evident that $\langle\langle\overline{\wp}\rangle\rangle = [[\wp]]$ and $[[\overline{\wp}]] = \langle\langle\wp\rangle\rangle$. As an example, let $\wp = [[x]]\langle\langle y\rangle\rangle\langle\langle z\rangle\rangle[[w]]\langle\langle v\rangle\rangle$. Then, we have $\langle\langle\wp\rangle\rangle = \{y, z, v\}$, $[[\wp]] = \{x, w\}$, $\mathrm{Dep}(\wp, x) = \mathrm{Dep}(\wp, w) = \emptyset$, $\mathrm{Dep}(\wp, y) = \mathrm{Dep}(\wp, z) = \{x\}$, $\mathrm{Dep}(\wp, v) = \{x, w\}$, and $\overline{\wp} = \langle\langle x\rangle\rangle[[y]][[z]]\langle\langle w\rangle\rangle[[v]]$.

Finally, we define the notion of *valuation* of variables over a generic set D, called *domain*, i.e., a partial function $\mathrm{v} : \mathrm{Var} \rightharpoonup \mathrm{D}$ mapping every variable in its domain to an element in D. By $\mathrm{Val_D}(\mathrm{V}) \triangleq \mathrm{V} \rightarrow \mathrm{D}$ we denote the set of all valuation functions over D defined on $\mathrm{V} \subseteq \mathrm{Var}$.

At this point, we give a general high-level semantics for the quantification prefixes by means of the following main definition of *dependence map*.

Definition 4.5 (*Dependence Maps*). Let $\wp \in \mathrm{Qnt}(\mathrm{V})$ be a quantification prefix over a set $\mathrm{V} \subseteq \mathrm{Var}$ of variables, and D a set. Then, a *dependence map* for $\wp$ over D is a function $\theta : \mathrm{Val_D}([[\wp]]) \rightarrow \mathrm{Val_D}(\mathrm{V})$ satisfying the following properties:

(1) $\theta(\mathrm{v})_{\restriction [[\wp]]} = \mathrm{v}$, for all $\mathrm{v} \in \mathrm{Val_D}([[\wp]])$; [9]
(2) $\theta(\mathrm{v}_1)(x) = \theta(\mathrm{v}_2)(x)$, for all $\mathrm{v}_1, \mathrm{v}_2 \in \mathrm{Val_D}([[\wp]])$ and $x \in \langle\langle\wp\rangle\rangle$ such that $\mathrm{v}_{1 \restriction \mathrm{Dep}(\wp, x)} = \mathrm{v}_{2 \restriction \mathrm{Dep}(\wp, x)}$.

$\mathrm{DM_D}(\wp)$ denotes the set of all dependence maps for $\wp$ over D.

Intuitively, Item 1 asserts that θ takes the same values of its argument w.r.t. the universal variables in $\wp$ and Item 2 ensures that the value of θ w.r.t. an existential variable x in $\wp$ does not depend on variables not in $\mathrm{Dep}(\wp, x)$. To get a better insight into this definition, a dependence map θ for $\wp$ can be considered as a set of *Skolem functions* that, given a value for each variable in $\mathrm{V}(\wp)$ that is universally quantified in $\wp$, returns a possible value for all the existential variables in $\wp$, in a way that is consistent w.r.t. the order of quantifications. Observe that, each $\theta \in \mathrm{DM_D}(\wp)$ is injective, so, $|\mathrm{rng}(\theta)| = |\mathrm{dom}(\theta)| = |\mathrm{D}|^{|[[\wp]]|}$. Moreover, $|\mathrm{DM_D}(\wp)| = \prod_{x \in \langle\langle\wp\rangle\rangle} |\mathrm{D}|^{|\mathrm{D}|^{|\mathrm{Dep}(\wp, x)|}}$. As an example, let $\mathrm{D} = \{0, 1\}$ and $\wp = [[x]]\langle\langle y\rangle\rangle[[z]] \in \mathrm{Qnt}(\mathrm{V})$ be a quantification prefix over $\mathrm{V} = \{x, y, z\}$. Then, we have that $|\mathrm{DM_D}(\wp)| = 4$ and $|\mathrm{DM_D}(\overline{\wp})| = 8$. Moreover, the dependence maps $\theta_i \in \mathrm{DM_D}(\wp)$ with $i \in [0, 3]$ and $\overline{\theta}_i \in \mathrm{DM_D}(\overline{\wp})$ with $i \in [0, 7]$, for a particular fixed order, are such that $\theta_0(\mathrm{v})(y) = 0$, $\theta_1(\mathrm{v})(y) = \mathrm{v}(x)$, $\theta_2(\mathrm{v})(y) = 1 - \mathrm{v}(x)$, and $\theta_3(\mathrm{v})(y) = 1$, for all $\mathrm{v} \in \mathrm{Val_D}([[\wp]])$, and $\overline{\theta}_i(\overline{\mathrm{v}})(x) = 0$ with $i \in [0, 3]$, $\overline{\theta}_i(\overline{\mathrm{v}})(x) = 1$ with $i \in [4, 7]$, $\overline{\theta}_0(\overline{\mathrm{v}})(z) = \overline{\theta}_4(\overline{\mathrm{v}})(z) = 0$, $\overline{\theta}_1(\overline{\mathrm{v}})(z) = \overline{\theta}_5(\overline{\mathrm{v}})(z) = \overline{\mathrm{v}}(y)$, $\overline{\theta}_2(\overline{\mathrm{v}})(z) = \overline{\theta}_6(\overline{\mathrm{v}})(z) = 1 - \overline{\mathrm{v}}(y)$, and $\overline{\theta}_3(\overline{\mathrm{v}})(z) = \overline{\theta}_7(\overline{\mathrm{v}})(z) = 1$, for all $\overline{\mathrm{v}} \in \mathrm{Val_D}([[\overline{\wp}]])$.

We now prove the following fundamental theorem that describes how to eliminate the strategy quantifications of an SL formula via a choice of a suitable dependence map over strategies. This

---

[9]By $\mathrm{g}_{\restriction Z} : (\mathrm{X} \cap \mathrm{Z}) \rightarrow \mathrm{Y}$ we denote the *restriction* of a function $\mathrm{g} : \mathrm{X} \rightarrow \mathrm{Y}$ to the elements in the set Z.

procedure can be seen as the equivalent of *Skolemization* in first order logic (see [Hodges 1993], for more details).

THEOREM 4.6 (SL STRATEGY QUANTIFICATION). *Let $\mathcal{G}$ be a CGS and $\varphi = \wp\psi$ an SL formula, being $\wp \in \mathrm{Qnt}(V)$ a quantification prefix over a set $V \subseteq \mathrm{free}(\psi) \cap \mathrm{Var}$ of variables. Then, for all assignments $\chi \in \mathrm{Asg}(\mathrm{free}(\varphi), s_0)$, the following holds: $\mathcal{G}, \chi, s_0 \models \varphi$ iff there exists a dependence map $\theta \in \mathrm{DM}_{\mathrm{Str}(s_0)}(\wp)$ such that $\mathcal{G}, \chi \uplus \theta(\chi'), s_0 \models \psi$, for all $\chi' \in \mathrm{Asg}([\![\wp]\!], s_0)$.* [10]

PROOF. The proof proceeds by induction on the length of the quantification prefix $\wp$. For the base case $|\wp| = 0$, the thesis immediately follows, since $[\![\wp]\!] = \emptyset$ and, consequently, both $\mathrm{DM}_{\mathrm{Str}(s_0)}(\wp)$ and $\mathrm{Asg}([\![\wp]\!], s_0)$ contain the empty function only (we are assuming, by convention, that $\varnothing(\varnothing) \triangleq \varnothing$).

We now prove, separately, the two directions of the inductive case.

[Only if]. Suppose that $\mathcal{G}, \chi, s_0 \models \varphi$, where $\wp = \mathrm{Qn} \cdot \wp'$. Then, two possible cases arise: either $\mathrm{Qn} = \langle\!\langle x \rangle\!\rangle$ or $\mathrm{Qn} = [\![x]\!]$.

— $\mathrm{Qn} = \langle\!\langle x \rangle\!\rangle$.
By Item 3a of Definition 2.16 of SL semantics, there is a strategy $\mathsf{f} \in \mathrm{Str}(s_0)$ such that $\mathcal{G}, \chi[x \mapsto \mathsf{f}], s_0 \models \wp'\psi$. Note that $[\![\wp]\!] = [\![\wp']\!]$. By the inductive hypothesis, we have that there exists a dependence map $\theta \in \mathrm{DM}_{\mathrm{Str}(s_0)}(\wp')$ such that $\mathcal{G}, \chi[x \mapsto \mathsf{f}] \uplus \theta(\chi'), s_0 \models \psi$, for all $\chi' \in \mathrm{Asg}([\![\wp']\!], s_0)$. Now, consider the function $\theta' : \mathrm{Asg}([\![\wp']\!], s_0) \to \mathrm{Asg}(V, s_0)$ defined by $\theta'(\chi') \triangleq \theta(\chi')[x \mapsto \mathsf{f}]$, for all $\chi' \in \mathrm{Asg}([\![\wp]\!], s_0)$. It is easy to check that θ' is a dependence map for $\wp$ over $\mathrm{Str}(s_0)$, i.e., $\theta' \in \mathrm{DM}_{\mathrm{Str}(s_0)}(\wp)$. Moreover, $\chi[x \mapsto \mathsf{f}] \uplus \theta(\chi') = \chi \uplus \theta(\chi')[x \mapsto \mathsf{f}] = \chi \uplus \theta'(\chi')$, for $\chi' \in \mathrm{Asg}([\![\wp]\!], s_0)$. Hence, $\mathcal{G}, \chi \uplus \theta'(\chi'), s_0 \models \psi$, for all $\chi' \in \mathrm{Asg}([\![\wp]\!], s_0)$.

— $\mathrm{Qn} = [\![x]\!]$.
By Item 3b of Definition 2.16, we have that, for all strategies $\mathsf{f} \in \mathrm{Str}(s_0)$, it holds that $\mathcal{G}, \chi[x \mapsto \mathsf{f}], s_0 \models \wp'\psi$. Note that $[\![\wp]\!] = [\![\wp']\!] \cup \{x\}$. By the inductive hypothesis, we derive that, for each $\mathsf{f} \in \mathrm{Str}(s_0)$, there exists a dependence map $\theta_{\mathsf{f}} \in \mathrm{DM}_{\mathrm{Str}(s_0)}(\wp')$ such that $\mathcal{G}, \chi[x \mapsto \mathsf{f}] \uplus \theta_{\mathsf{f}}(\chi'), s_0 \models \psi$, for all $\chi' \in \mathrm{Asg}([\![\wp']\!], s_0)$. Now, consider the function $\theta' : \mathrm{Asg}([\![\wp]\!], s_0) \to \mathrm{Asg}(V, s_0)$ defined by $\theta'(\chi') \triangleq \theta_{\chi'(x)}(\chi'_{\upharpoonright[\![\wp']\!]})[x \mapsto \chi'(x)]$, for all $\chi' \in \mathrm{Asg}([\![\wp]\!], s_0)$. It is evident that θ' is a dependence map for $\wp$ over $\mathrm{Str}(s_0)$, i.e., $\theta' \in \mathrm{DM}_{\mathrm{Str}(s_0)}(\wp)$. Moreover, $\chi[x \mapsto \mathsf{f}] \uplus \theta_{\mathsf{f}}(\chi') = \chi \uplus \theta_{\mathsf{f}}(\chi')[x \mapsto \mathsf{f}] = \chi \uplus \theta'(\chi'[x \mapsto \mathsf{f}])$, for $\mathsf{f} \in \mathrm{Str}(s_0)$ and $\chi' \in \mathrm{Asg}([\![\wp']\!], s_0)$. Hence, $\mathcal{G}, \chi \uplus \theta'(\chi'), s_0 \models \psi$, for all $\chi' \in \mathrm{Asg}([\![\wp]\!], s_0)$.

[If]. Suppose that there exists a dependence map $\theta \in \mathrm{DM}_{\mathrm{Str}(s_0)}(\wp)$ such that $\mathcal{G}, \chi \uplus \theta(\chi'), s_0 \models \psi$, for all $\chi' \in \mathrm{Asg}([\![\wp]\!], s_0)$, where $\wp = \mathrm{Qn} \cdot \wp'$. Then, two possible cases arise: either $\mathrm{Qn} = \langle\!\langle x \rangle\!\rangle$ or $\mathrm{Qn} = [\![x]\!]$.

— $\mathrm{Qn} = \langle\!\langle x \rangle\!\rangle$.
There is a strategy $\mathsf{f} \in \mathrm{Str}(s_0)$ such that $\mathsf{f} = \theta(\chi')(x)$, for all $\chi' \in \mathrm{Asg}([\![\wp]\!], s_0)$. Note that $[\![\wp]\!] = [\![\wp']\!]$. Consider the function $\theta' : \mathrm{Asg}([\![\wp']\!], s_0) \to \mathrm{Asg}(V \setminus \{x\}, s_0)$ defined by $\theta'(\chi') \triangleq \theta(\chi')_{\upharpoonright(V \setminus \{x\})}$, for all $\chi' \in \mathrm{Asg}([\![\wp']\!], s_0)$. It is easy to check that θ' is a dependence map for $\wp'$ over $\mathrm{Str}(s_0)$, i.e., $\theta' \in \mathrm{DM}_{\mathrm{Str}(s_0)}(\wp')$. Moreover, $\chi \uplus \theta(\chi') = \chi \uplus \theta'(\chi')[x \mapsto \mathsf{f}] = \chi[x \mapsto \mathsf{f}] \uplus \theta'(\chi')$, for $\chi' \in \mathrm{Asg}([\![\wp']\!], s_0)$. Then, it is evident that $\mathcal{G}, \chi[x \to \mathsf{f}] \uplus \theta'(\chi'), s_0 \models \psi$, for all $\chi' \in \mathrm{Asg}([\![\wp']\!], s_0)$. By the inductive hypothesis, we derive that $\mathcal{G}, \chi[x \mapsto \mathsf{f}], s_0 \models \wp'\psi$, which means that $\mathcal{G}, \chi, s_0 \models \varphi$, by Item 3a of Definition 2.16 of SL semantics.

— $\mathrm{Qn} = [\![x]\!]$.
First note that $[\![\wp]\!] = [\![\wp']\!] \cup \{x\}$. Also, consider the functions $\theta'_{\mathsf{f}} : \mathrm{Asg}([\![\wp']\!], s_0) \to \mathrm{Asg}(V \setminus \{x\}, s_0)$ defined by $\theta'_{\mathsf{f}}(\chi') \triangleq \theta(\chi'[x \mapsto \mathsf{f}])_{\upharpoonright(V \setminus \{x\})}$, for each $\mathsf{f} \in \mathrm{Str}(s_0)$ and $\chi' \in \mathrm{Asg}([\![\wp']\!], s_0)$.

---

[10] By $\mathsf{g}_1 \uplus \mathsf{g}_2 : (X_1 \cup X_2) \to (Y_1 \cup Y_2)$ we denote the operation of *union* of two functions $\mathsf{g}_1 : X_1 \to Y_1$ and $\mathsf{g}_2 : X_2 \to Y_2$ defined on disjoint domains, i.e., $X_1 \cap X_2 = \emptyset$.

It is easy to see that every θ'_f is a dependence map for $\wp'$ over $\mathrm{Str}(s_0)$, i.e., $\theta'_f \in \mathrm{DM}_{\mathrm{Str}(s_0)}(\wp')$. Moreover, $\chi \uplus \theta(\chi') = \chi \uplus \theta'_{\chi'(x)}(\chi'_{\upharpoonright[\![\wp']\!]})[x \mapsto \chi'(x)] = \chi[x \mapsto \chi'(x)] \uplus \theta'_{\chi'(x)}(\chi'_{\upharpoonright[\![\wp']\!]})$, for $\chi' \in \mathrm{Asg}([\![\wp]\!], s_0)$. Then, it is evident that $\mathcal{G}, \chi[x \to f] \uplus \theta'_f(\chi'), s_0 \models \psi$, for all $f \in \mathrm{Str}(s_0)$ and $\chi' \in \mathrm{Asg}([\![\wp']\!], s_0)$. By the inductive hypothesis, we derive that $\mathcal{G}, \chi[x \mapsto f], s_0 \models \wp'\psi$, for all $f \in \mathrm{Str}(s_0)$, which means that $\mathcal{G}, \chi, s_0 \models \varphi$, by Item 3b of Definition 2.16.

Thus, the thesis of the theorem holds. $\quad\square$

As an immediate consequence of the previous result, we derive the following corollary.

COROLLARY 4.7 (SL STRATEGY QUANTIFICATION). *Let $\mathcal{G}$ be a CGS and $\varphi = \wp\psi$ an SL sentence, where ψ is agent-closed and $\wp \in \mathrm{Qnt}(\mathrm{free}(\psi))$. Then, $\mathcal{G} \models \varphi$ iff there exists a dependence map $\theta \in \mathrm{DM}_{\mathrm{Str}(s_0)}(\wp)$ such that $\mathcal{G}, \theta(\chi), s_0 \models \psi$, for all $\chi \in \mathrm{Asg}([\![\wp]\!], s_0)$.*

4.3. Elementary quantifications

We now have all tools we need to introduce the property of elementariness for a particular class of dependence maps. Intuitively, a dependence map over functions from a set T to a set D is elementary if it can be split into a set of dependence maps over D, one for each element of T. This idea allows us to enormously simplify the reasoning about strategy quantifications, since we can reduce them to a set of quantifications over actions, one for each track in their domains. This means that, under certain conditions, we can transform a dependence map $\theta \in \mathrm{DM}_{\mathrm{Str}(s)}(\wp)$ over strategies in a function $\widetilde{\theta} : \mathrm{Trk}(s) \to \mathrm{DM}_{\mathrm{Ac}}(\wp)$ that associates with each track a dependence map over actions.

To formally develop the above idea, we have first to introduce the generic concept of adjoint function and state an auxiliary lemma.

Definition 4.8 (*Adjoint Functions*). Let D, T, U, and V be four sets, and $m : (T \to D)^U \to (T \to D)^V$ and $\widetilde{m} : T \to (D^U \to D^V)$ two functions. Then, $\widetilde{m}$ is the *adjoint* of m if $\widetilde{m}(t)(\widehat{g}(t))(x) = m(g)(x)(t)$, for all $g \in (T \to D)^U$, $x \in V$, and $t \in T$ [11]

Intuitively, $\widetilde{m}$ is the adjoint of m if the dependence from the set T in both domain and codomain of the latter can be extracted and put as a common factor of the functor given by the former. This means also that, for every pair of functions $g_1, g_2 \in (T \to D)^U$ such that $\widehat{g_1}(t) = \widehat{g_2}(t)$ for some $t \in T$, it holds that $m(g_1)(x)(t) = m(g_2)(x)(t)$, for all $x \in V$. It is immediate to observe that if a function has an adjoint then this adjoint is unique. At the same way, if one has an adjoint function then it is possible to determine the original function without any ambiguity. Thus, it is established a one-to-one correspondence between functions admitting an adjoint and the adjoint itself.

Next lemma formally states the property briefly described above, i.e., that each dependence map over a set $T \to D$, admitting an adjoint function, can be represented as a function, with T as domain, which returns dependence maps over D as values.

LEMMA 4.9 (ADJOINT DEPENDENCE MAPS). *Let $\wp \in \mathrm{Qnt}(V)$ be a quantification prefix over a set $V \subseteq \mathrm{Var}$ of variables, D and T two sets, and $\theta : \mathrm{Val}_{T \to D}([\![\wp]\!]) \to \mathrm{Val}_{T \to D}(V)$ and $\widetilde{\theta} : T \to (\mathrm{Val}_D([\![\wp]\!]) \to \mathrm{Val}_D(V))$ two functions such that $\widetilde{\theta}$ is the adjoint of θ. Then, $\theta \in \mathrm{DM}_{T \to D}(\wp)$ iff, for all $t \in T$, it holds that $\widetilde{\theta}(t) \in \mathrm{DM}_D(\wp)$.*

We now define the formal meaning of the elementariness of a dependence map over functions.

Definition 4.10 (*Elementary Dependence Maps*). Let $\wp \in \mathrm{Qnt}(V)$ be a quantification prefix over a set $V \subseteq \mathrm{Var}$ of variables, D and T two sets, and $\theta \in \mathrm{DM}_{T \to D}(\wp)$ a dependence map for $\wp$ over $T \to D$. Then, θ is *elementary* if it admits an adjoint function. $\mathrm{EDM}_{T \to D}(\wp)$ denotes the set of all elementary dependence maps for $\wp$ over $T \to D$.

---

[11] By $\widehat{g} : Y \to X \to Z$ we denote the operation of *flipping* of a function $g : X \to Y \to Z$.

It is important to observe that, unfortunately, there are dependence maps that are not elementary. To easily understand why this is actually the case, it is enough to count both the number of dependence maps $\mathrm{DM}_{\mathrm{T}\to\mathrm{D}}(\wp)$ and of adjoint functions $\mathrm{T} \to \mathrm{DM}_{\mathrm{D}}(\wp)$, where $|\mathrm{D}| > 1$, $|\mathrm{T}| > 1$ and $\wp$ is such that there is an $x \in \langle\!\langle\wp\rangle\!\rangle$ for which $\mathrm{Dep}(\wp, x) \neq \emptyset$. Indeed, it holds that $|\mathrm{DM}_{\mathrm{T}\to\mathrm{D}}(\wp)| = \prod_{x\in\langle\!\langle\wp\rangle\!\rangle} |\mathrm{D}|^{|\mathrm{T}|\cdot|\mathrm{D}|^{|\mathrm{T}|\cdot|\mathrm{Dep}(\wp,x)|}} > \prod_{x\in\langle\!\langle\wp\rangle\!\rangle} |\mathrm{D}|^{|\mathrm{T}|\cdot|\mathrm{D}|^{|\mathrm{Dep}(\wp,x)|}} = |\mathrm{T} \to \mathrm{DM}_{\mathrm{D}}(\wp)|$. So, there are much more dependence maps, a number double exponential in $|\mathrm{T}|$, than possible adjoint functions, whose number is only exponential in this value. Furthermore, observe that the simple set $\mathrm{Qnt}_{\exists^*\forall^*}(\mathrm{V}) \triangleq \{\wp \in \mathrm{Qnt}(\mathrm{V}) : \exists i \in [0, |\wp|] \,.\, [\![(\wp)_{<i}]\!] = \emptyset \wedge \langle\!\langle(\wp)_{\geq i}\rangle\!\rangle = \emptyset\}$, for $\mathrm{V} \subseteq \mathrm{Var}$, is the maximal class of quantification prefixes that admits only elementary dependence maps over $\mathrm{T} \to \mathrm{D}$, i.e., it is such that each $\theta \in \mathrm{DM}_{\mathrm{T}\to\mathrm{D}}(\wp)$ is elementary, for all $\wp \in \mathrm{Qnt}_{\exists^*\forall^*}(\mathrm{V})$. This is due to the fact that there are no functional dependences between variables, i.e., for each $x \in \langle\!\langle\wp\rangle\!\rangle$, it holds that $\mathrm{Dep}(\wp, x) = \emptyset$.

Finally, we can introduce a new very important semantics for SL syntactic fragments, which is based on the concept of elementary dependence map over strategies, and we refer to the related satisfiability concept as *elementary satisfiability*, in symbols $\models_{\mathrm{E}}$. Intuitively, such a semantics has the peculiarity that a strategy, used in an existential quantification in order to satisfy a formula, is only chosen between those that are elementary w.r.t. the universal quantifications. In this way, when we have to decide what is its value c on a given track ρ, we do it only in dependence of the values on the same track of the strategies so far quantified, but not on their whole structure, as it is the case instead of the classic semantics. This means that c does not depend on the values of the other strategies on tracks ρ' that extend ρ, i.e., it does not depend on future choices made on ρ'. In addition, we have that c does not depend on values on parallel tracks ρ' that only share a prefix with ρ, i.e., it is independent on choices made on the possibly alternative futures ρ'. The elementary semantics of SL[NG] formulas involving atomic propositions, Boolean connectives, temporal operators, and agent bindings is defined as for the classic one, where the modeling relation $\models$ is substituted with $\models_{\mathrm{E}}$, and we omit to report it here. In the following definition, we only describe the part concerning the quantification prefixes.

Definition 4.11 (SL[NG] *Elementary Semantics*). Let $\mathcal{G}$ be a CGS, $s \in \mathrm{St}$ one of its states, and $\wp\psi$ an SL[NG] formula, where ψ is agent-closed and $\wp \in \mathrm{Qnt}(\mathrm{free}(\psi))$. Then $\mathcal{G}, \varnothing, s \models_{\mathrm{E}} \wp\psi$ if there is an elementary dependence map $\theta \in \mathrm{EDM}_{\mathrm{Str}(s)}(\wp)$ for $\wp$ over $\mathrm{Str}(s)$ such that $\mathcal{G}, \theta(\chi), s \models_{\mathrm{E}} \psi$, for all $\chi \in \mathrm{Asg}([\![\wp]\!], s)$.

It is immediate to see a strong similarity between the statement of Corollary 4.7 of SL strategy quantification and the previous definition. The only crucial difference resides in the choice of the kind of dependence map. Moreover, observe that, differently from the classic semantics, the quantifications in the prefix are not treated individually but as an atomic block. This is due to the necessity of having a strict correlation between the point-wise structure of the quantified strategies.

Remark 4.12 (SL *Elementary Semantics*). It can be interesting to know that we do not define an elementary semantics for the whole SL, since we are not able, at the moment, to easily use the concept of elementary dependence map, when the quantifications are not necessarily grouped in prefixes, i.e., when the formula is not in prenex normal form. In fact, this may represent a challenging problem, whose solution is left to future works.

Due to the new semantics of SL[NG], we have to redefine the related concepts of model and satisfiability, in order to differentiate between the classic relation $\models$ and the elementary one $\models_{\mathrm{E}}$. Indeed, as we show later, there are sentences that are satisfiable but not elementary satisfiable and vice versa.

Definition 4.13 (SL[NG] *Elementary Satisfiability*). We say that a CGS $\mathcal{G}$ is an *elementary model* of an SL[NG] sentence φ, in symbols $\mathcal{G} \models_{\mathrm{E}} \varphi$, if $\mathcal{G}, \varnothing, s_0 \models_{\mathrm{E}} \varphi$. In general, we also say

that $\mathcal{G}$ is a *elementary model* for φ on $s \in \mathrm{St}$, in symbols $\mathcal{G}, s \models_{\mathrm{E}} \varphi$, if $\mathcal{G}, \varnothing, s \models_{\mathrm{E}} \varphi$. An SL[NG] sentence φ is *elementarily satisfiable* if there is an elementary model for it.

We have to modify the concepts of implication and equivalence, as well. Indeed, also in this case we can have pairs of equivalent formulas that are not elementarily equivalent, and vice versa. Thus, we have to be careful when we use natural transformation between formulas, since it can be the case that they preserve the meaning only under the classic semantics. An example of this problem can arise when one want to put a formula in *pnf*.

Definition 4.14 (SL[NG] *Elementary Implication and Equivalence*). Given two SL[NG] formulas φ_1 and φ_2 with free(φ_1) = free(φ_2), we say that φ_1 *elementarily implies* φ_2, in symbols $\varphi_1 \Rightarrow_{\mathrm{E}} \varphi_2$, if, for all CGSs $\mathcal{G}$, states $s \in \mathrm{St}$, and free(φ_1)-defined s-total assignments $\chi \in \mathrm{Asg}(\mathrm{free}(\varphi_1), s)$, it holds that if $\mathcal{G}, \chi, s \models_{\mathrm{E}} \varphi_1$ then $\mathcal{G}, \chi, s \models_{\mathrm{E}} \varphi_2$. Accordingly, we say that φ_1 is *elementarily equivalent* to φ_2, in symbols $\varphi_1 \equiv_{\mathrm{E}} \varphi_2$, if both $\varphi_1 \Rightarrow_{\mathrm{E}} \varphi_2$ and $\varphi_2 \Rightarrow_{\mathrm{E}} \varphi_1$ hold.

4.4. Elementariness and non-elementariness

Finally, we show that the introduced concept of elementary satisfiability is relevant to the context of our logic, as its applicability represents a demarcation line between "easy" and "hard" fragments of SL. Moreover, we believe that it is because of this fundamental property that several well-known temporal logics are so robustly decidable [Vardi 1996].

Remark 4.15 (SL[NG, 0-alt] *Elementariness*). It is interesting to observe that, for every CGS $\mathcal{G}$ and SL[NG, 0-alt] sentence φ, it holds that $\mathcal{G} \models \varphi$ iff $\mathcal{G} \models_{\mathrm{E}} \varphi$. This is an immediate consequence of the fact that all quantification prefixes $\wp$ used in φ belong to $\mathrm{Qnt}_{\exists^* \forall^*}(\mathrm{V})$, for a given set $\mathrm{V} \subseteq \mathrm{Var}$ of variables. Thus, as already mentioned, the related dependence maps on strategies $\theta \in \mathrm{DM}_{\mathrm{Str}(s_0)}(\wp)$ are necessarily elementary.

By Corollary 4.7 of SL strategy quantification, it is easy to see that the following coherence property about the elementariness of the SL[NG] satisfiability holds. Intuitively, it asserts that every elementarily satisfiable sentence in *pnf* is satisfiable too.

THEOREM 4.16 (SL[NG] ELEMENTARY COHERENCE). *Let $\mathcal{G}$ be a* CGS, $s \in \mathrm{St}$ *one of its states, φ an* SL[NG] *formula in pnf, and $\chi \in \mathrm{Asg}(s)$ an s-total assignment with* free$(\varphi) \subseteq$ dom(χ). *Then, it holds that $\mathcal{G}, \chi, s \models_{\mathrm{E}} \varphi$ implies $\mathcal{G}, \chi, s \models \varphi$.*

PROOF. The proof proceeds by induction on the structure of the formula. For the sake of succinctness, we only show the crucial case of principal subsentences $\phi \in \mathrm{psnt}(\varphi)$, i.e., when ϕ is of the form $\wp \psi$, where $\wp \in \mathrm{Qnt}(\mathrm{free}(\psi))$ is a quantification prefix, and ψ is an agent-closed formula.

Suppose that $\mathcal{G}, \varnothing, s \models_{\mathrm{E}} \wp \psi$. Then, by Definition 4.11 of SL[NG] elementary semantics, there is an elementary dependence map $\theta \in \mathrm{EDM}_{\mathrm{Str}(s)}(\wp)$ such that, for all assignments $\chi \in \mathrm{Asg}(\llbracket \wp \rrbracket, s)$, it holds that $\mathcal{G}, \theta(\chi), s \models_{\mathrm{E}} \psi$. Now, by the inductive hypothesis, there is a dependence map $\theta \in \mathrm{DM}_{\mathrm{Str}(s)}(\wp)$ such that, for all assignments $\chi \in \mathrm{Asg}(\llbracket \wp \rrbracket, s)$, it holds that $\mathcal{G}, \theta(\chi), s \models \psi$. Hence, by Corollary 4.7 of SL strategy quantification, we have that $\mathcal{G}, \varnothing, s \models \wp \psi$. □

However, it is worth noting that the converse property may not hold, as we show in the next theorem, i.e., there are sentences in *pnf* that are satisfiable but not elementarily satisfiable. Note that the following results already holds for CHP-SL.

THEOREM 4.17 (TB-SL[BG] NON-ELEMENTARINESS). *There exists a satisfiable* TB-SL[BG, 1-ag, 2-var, 1-alt] *sentence in pnf that is not elementarily satisfiable.*

PROOF. Consider the TB-SL[BG, 1-ag, 2-var, 1-alt] sentence $\varphi \triangleq \varphi_1 \wedge \varphi_2$ in *pnf* where $\varphi_1 \triangleq \wp(\psi_1 \wedge \psi_2)$, with $\wp \triangleq \llbracket x \rrbracket \langle\langle y \rangle\rangle$, $\psi_1 \triangleq (\alpha, x)\mathsf{X}\, \mathsf{p} \leftrightarrow (\alpha, y)\mathsf{X}\, \neg\mathsf{p}$, and $\psi_2 \triangleq (\alpha, x)\mathsf{X}\,\mathsf{X}\,\mathsf{p} \leftrightarrow (\alpha, y)\mathsf{X}\,\mathsf{X}\,\mathsf{p}$, and $\varphi_2 \triangleq \llbracket x \rrbracket (\alpha, x)\mathsf{X}\,(((\langle\langle x \rangle\rangle(\alpha, x)\mathsf{X}\,\mathsf{p}) \wedge (\langle\langle x \rangle\rangle(\alpha, x)\mathsf{X}\,\neg\mathsf{p}))$. Moreover, note that the

TB-SL[1G, 1-ag, 1-var, 0-alt] sentence φ_2 is equivalent to the CTL formula AX $((EX\ p) \wedge (EX\ \neg p))$. Then, it is easy to see that the turn-based CGS $\mathcal{G}_{Rdc}$ of Figure 4 on page 15 satisfies φ. Indeed, $\mathcal{G}_{Rdc}, \theta(\chi), s_0 \models \psi_1 \wedge \psi_2$, for all assignments $\chi \in \mathrm{Asg}(\{x\}, s_0)$, where the non-elementary dependence map $\theta \in \mathrm{DM}_{\mathrm{Str}(s_0)}(\wp)$ is such that $\theta(\chi)(y)(s_0) = \neg\chi(x)(s_0)$ and $\theta(\chi)(y)(s_0 \cdot s_i) = \chi(x)(s_0 \cdot s_{1-i})$, for all $i \in \{0, 1\}$.

Now, let $\mathcal{G}$ be a generic CGS. If $\mathcal{G} \not\models \varphi$, by Theorem 4.16 of SL[NG] elementary coherence, it holds that $\mathcal{G} \not\models_E \varphi$. Otherwise, we have that $\mathcal{G} \models \varphi$ and, in particular, $\mathcal{G} \models \varphi_1$, which means that $\mathcal{G} \models \wp(\psi_1 \wedge \psi_2)$. At this point, to prove that $\mathcal{G} \not\models_E \varphi$, we show that, for all elementary dependence maps $\theta \in \mathrm{EDM}_{\mathrm{Str}(s_0)}(\wp)$, there exists an assignment $\chi \in \mathrm{Asg}(\{x\}, s_0)$ such that $\mathcal{G}, \theta(\chi), s_0 \not\models_E \psi_1 \wedge \psi_2$. To do this, let us fix an elementary dependence map θ and an assignment χ. Also, assume $s_1 \triangleq \tau(s_0, \emptyset[\alpha \mapsto \chi(x)(s_0)])$ and $s_2 \triangleq \tau(s_0, \emptyset[\alpha \mapsto \theta(\chi)(y)(s_0)])$. Now, we distinguish between two cases.

— $p \in \lambda(s_1)$ iff $p \in \lambda(s_2)$. In this case, we can easily observe that $\mathcal{G}, \theta(\chi), s_0 \not\models \psi_1$ and consequently, by Theorem 4.16, it holds that $\mathcal{G}, \theta(\chi), s_0 \not\models_E \psi_1 \wedge \psi_2$. So, we are done.
— $p \in \lambda(s_1)$ iff $p \notin \lambda(s_2)$. If $\mathcal{G}, \theta(\chi), s_0 \not\models \psi_2$ then, by Theorem 4.16, it holds that $\mathcal{G}, \theta(\chi), s_0 \not\models_E \psi_1 \wedge \psi_2$. So, we are done. Otherwise, let $s_3 \triangleq \tau(s_1, \emptyset[\alpha \mapsto \chi(x)(s_0 \cdot s_1)])$ and $s_4 \triangleq \tau(s_2, \emptyset[\alpha \mapsto \theta(\chi)(y)(s_0 \cdot s_2)])$. Then, it holds that $p \in \lambda(s_3)$ iff $p \in \lambda(s_4)$. Now, consider a new assignment $\chi' \in \mathrm{Asg}(\{x\}, s_0)$ such that $\chi'(x)(s_0 \cdot s_2) = \chi(x)(s_0 \cdot s_2)$ and $p \in \lambda(s_3')$ iff $p \notin \lambda(s_4)$, where $s_3' \triangleq \tau(s_1, \emptyset[\alpha \mapsto \chi'(x)(s_0 \cdot s_1)])$. Observe that the existence of such an assignment, with particular reference to the second condition, is ensured by the fact that $\mathcal{G} \models \varphi_2$. At this point, due to the elementariness of the dependence map θ, we have that $\theta(\chi')(y)(s_0 \cdot s_2) = \theta(\chi)(y)(s_0 \cdot s_2)$. Consequently, it holds that $s_4 = \tau(s_2, \emptyset[\alpha \mapsto \theta(\chi')(y)(s_0 \cdot s_2)])$. Thus, $\mathcal{G}, \theta(\chi'), s_0 \not\models \psi_2$, which implies, by Theorem 4.16, that $\mathcal{G}, \theta(\chi'), s_0 \not\models_E \psi_1 \wedge \psi_2$. So, we are done.

Thus, the thesis of the theorem holds. □

The following corollary is an immediate consequence of the previous theorem. It is interesting to note that, at the moment, we do not know if such a result can be extended to the simpler GL fragment.

COROLLARY 4.18 (SL[BG] NON-ELEMENTARINESS). *There exists a satisfiable* SL[BG, 1-ag, 2-var, 1-alt] *sentence in pnf that is not elementarily satisfiable.*

Remark 4.19 (*Kinds of Non-Elementariness*).   It is worth remarking that the kind of non-elementariness of the sentence φ shown in the above theorem can be called *essential*, i.e., it cannot be eliminated, due to the fact that φ is satisfiable but not elementarily satisfiable. However, there are different sentences, such as the conjunct φ_1 in φ, having both models on which they are elementarily satisfiable and models, like the CGS $\mathcal{G}_{Rdc}$, on which they are only non-elementarily satisfiable. Such a kind of non-elementariness can be called *non-essential*, since it can be eliminated by an opportune choice of the underlying model. Note that a similar reasoning can be done for the dual concept of elementariness, which we call *essential* if all models satisfying a given sentence elementarily satisfy it as well.

Before continuing, we want to show the reason why we have redefined the concepts of implication and equivalence in the context of elementary semantics. Consider the SL[BG, 1-ag, 2-var, 1-alt] sentence φ_1 used in Theorem 4.17 of TB-SL[BG] non-elementariness. It is not hard to see that it is equivalent to the SL[1G, 1-ag, 1-var, 0-alt] $\varphi' \triangleq (\langle\!\langle x \rangle\!\rangle(\alpha, x)\psi_1 \leftrightarrow \langle\!\langle x \rangle\!\rangle(\alpha, x)\psi_2) \wedge (\langle\!\langle x \rangle\!\rangle(\alpha, x)\psi_3 \leftrightarrow \langle\!\langle x \rangle\!\rangle(\alpha, x)\psi_4)$, where $\psi_1 \triangleq X (p \wedge X p)$, $\psi_2 \triangleq X (\neg p \wedge X p)$, $\psi_3 \triangleq X (p \wedge X \neg p)$, and $\psi_4 \triangleq X (\neg p \wedge X \neg p)$. Note that φ' is in turn equivalent to the CTL* formula $(E\psi_1 \leftrightarrow E\psi_2) \wedge (E\psi_3 \leftrightarrow E\psi_4)$. However, φ_1 and φ' are not elementarily equivalent, since we have that $\mathcal{G}_{Rdc} \not\models_E \varphi_1$ but $\mathcal{G}_{Rdc} \models_E \varphi'$, where $\mathcal{G}_{Rdc}$ is the CGS of Figure 4 on page 15.

At this point, we can proceed with the proof of the elementariness of satisfiability for $\text{SL}[1\text{G}]$. It is important to note that there is no gap, in our knowledge, between the logics that are elementarily satisfiable and those that are not, since the fragment $\text{SL}[\text{BG}, 1\text{-ag}, 2\text{-var}, 1\text{-alt}]$ used in the previous theorem cannot be further reduced, due to the fact that otherwise it collapses into $\text{SL}[1\text{G}]$. Before starting, we have to describe some notation regarding classic two-player games on infinite words [Perrin and Pin 2004], which are used here as a technical tool. Note that we introduce the names of scheme and match in place of the more usual strategy and play, in order to avoid confusion between the concepts related to a CGS and those related to the tool.

A *two-player arena* (TPA, for short) is a tuple $\mathcal{A} \triangleq \langle \text{N}_\text{e}, \text{N}_\text{o}, E, n_0 \rangle$, where N_e and N_o are non-empty non-intersecting sets of *nodes* for player *even* and *odd*, respectively, $E \triangleq E_e \cup E_o$, with $E_e \subseteq \text{N}_\text{e} \times \text{N}_\text{o}$ and $E_o \subseteq \text{N}_\text{o} \times \text{N}_\text{e}$, is the *edge relation* between nodes, and $n_0 \in \text{N}_\text{o}$ is a designated *initial node*.

An *even position* in $\mathcal{A}$ is a finite non-empty sequence of nodes $\varrho \in \text{N}_\text{e}^+$ such that $(\varrho)_0 = n_0$ and, for all $i \in [0, |\varrho| - 1[$, there exists a node $n \in \text{N}_\text{o}$ for which $((\varrho)_i, n) \in E_e$ and $(n, (\varrho)_{i+1}) \in E_o$ hold. In addition, an *odd position* in $\mathcal{A}$ is a finite non-empty sequence of nodes $\varrho = \varrho' \cdot n \in \text{N}_\text{e}^+ \cdot \text{N}_\text{o}$, with $n \in \text{N}_\text{o}$, such that ϱ' is an even position and $(\text{lst}(\varrho'), n) \in E_e$. By Pos_e and Pos_o we denote, respectively, the sets of even and odd positions.

An *even* (resp., *odd*) *scheme* in $\mathcal{A}$ is a function $\text{s}_\text{e} : \text{Pos}_\text{e} \to \text{N}_\text{o}$ (resp., $\text{s}_\text{o} : \text{Pos}_\text{o} \to \text{N}_\text{e}$) that maps each even (resp., odd) position to an odd (resp., even) node in a way that is compatible with the edge relation E_e (resp., E_o), i.e., for all $\varrho \in \text{Pos}_\text{e}$ (resp., $\varrho \in \text{Pos}_\text{o}$), it holds that $(\text{lst}(\varrho), \text{s}_\text{e}(\varrho)) \in E_e$ (resp., $(\text{lst}(\varrho), \text{s}_\text{o}(\varrho)) \in E_o$). By Sch_e (resp., Sch_o) we indicate the sets of even (resp., odd) schemes.

A *match* in $\mathcal{A}$ is an infinite sequence of nodes $\varpi \in \text{N}_\text{e}^\omega$ such that $(\varpi)_0 = n_0$ and, for all $i \in \mathbb{N}$, there exists a node $n \in \text{N}_\text{o}$ such that $((\varpi)_i, n) \in E_e$ and $(n, (\varpi)_{i+1}) \in E_o$. By Mtc we denote the set of all matches. A *match map* $\text{mtc} : \text{Sch}_\text{e} \times \text{Sch}_\text{o} \to \text{Mtc}$ is a function that, given two schemes $\text{s}_\text{e} \in \text{Sch}_\text{e}$ and $\text{s}_\text{o} \in \text{Sch}_\text{o}$, returns the unique match $\varpi = \text{mtc}(\text{s}_\text{e}, \text{s}_\text{o})$ such that, for all $i \in \mathbb{N}$, it holds that $(\varpi)_{i+1} = \text{s}_\text{o}((\varpi)_{\leq i} \cdot \text{s}_\text{e}((\varpi)_{\leq i}))$.

A *two-player game* (TPG, for short) is a tuple $\mathcal{H} \triangleq \langle \mathcal{A}, \text{Win} \rangle$, where $\mathcal{A}$ is a TPA and $\text{Win} \subseteq \text{Mtc}$. On one hand, we say that player even wins $\mathcal{H}$ if there exists an even scheme $\text{s}_\text{e} \in \text{Sch}_\text{e}$ such that, for all odd schemes $\text{s}_\text{o} \in \text{Sch}_\text{o}$, it holds that $\text{mtc}(\text{s}_\text{e}, \text{s}_\text{o}) \in \text{Win}$. On the other hand, we say that player odd wins $\mathcal{H}$ if there exists an odd scheme $\text{s}_\text{o} \in \text{Sch}_\text{o}$ such that, for all even schemes $\text{s}_\text{e} \in \text{Sch}_\text{e}$, it holds that $\text{mtc}(\text{s}_\text{e}, \text{s}_\text{o}) \notin \text{Win}$.

In the following, for a given binding prefix $\flat \in \text{Bnd}(\text{V})$ with $\text{V} \subseteq \text{Var}$, we denote by $\zeta_\flat : \text{Ag} \to \text{V}$ the function associating with each agent the related variable in $\flat$, i.e., for all $a \in \text{Ag}$, there is $i \in [0, |\flat|[$ such that $(\flat)_i = (a, \zeta_\flat(a))$.

As first step towards the proof of the elementariness of $\text{SL}[1\text{G}]$, we have to give a construction of a two-player game, based on an a priori chosen CGS, in which the players are explicitly viewed one as a dependence map and the other as a valuation, both over actions. This construction results to be a deep technical evolution of the proof method used for the dualization of alternating automata on infinite objects [Muller and Schupp 1987].

Definition 4.20 (*Dependence-vs-Valuation Game*). Let $\mathcal{G}$ be a CGS, $s \in \text{St}$ one of its states, $\text{P} \subseteq \text{Pth}(s)$ a set of paths, $\wp \in \text{Qnt}(\text{V})$ a quantification prefix over a set $\text{V} \subseteq \text{Var}$ of variables, and $\flat \in \text{Bnd}(\text{V})$ a binding. Then, the dependence-vs-valuation game for $\mathcal{G}$ in s over P w.r.t. $\wp$ and $\flat$ is the TPG $\mathcal{H}(\mathcal{G}, s, \text{P}, \wp, \flat) \triangleq \langle \mathcal{A}(\mathcal{G}, s, \wp, \flat), \text{P} \rangle$, where the TPA $\mathcal{A}(\mathcal{G}, s, \wp, \flat) \triangleq \langle \text{St}, \text{St} \times \text{DM}_{\text{Ac}}(\wp), E, s \rangle$ has the edge relations defined as follows:

— $E_e \triangleq \{(t, (t, \theta)) : t \in \text{St} \wedge \theta \in \text{DM}_{\text{Ac}}(\wp)\}$;
— $E_o \triangleq \{((t, \theta), \tau(t, \theta(\text{v}) \circ \zeta_\flat)) : t \in \text{St} \wedge \theta \in \text{DM}_{\text{Ac}}(\wp) \wedge \text{v} \in \text{Val}_{\text{Ac}}(\llbracket \wp \rrbracket)\}$ [12].

---

[12]By $\text{g}_2 \circ \text{g}_1 : \text{X} \to \text{Z}$ we denote the operation of *composition* of two functions $\text{g}_1 : \text{X} \to \text{Y}_1$ and $\text{g}_2 : \text{Y}_2 \to \text{Z}$ with $\text{Y}_1 \subseteq \text{Y}_2$.

In the next lemma we state a fundamental relationship between dependence-vs-valuation games and their duals. Basically, we prove that if a player wins the game then the opposite player can win the dual game, and vice versa. This represents one of the two crucial steps in our elementariness proof.

LEMMA 4.21 (DEPENDENCE-VS-VALUATION DUALITY). *Let $\mathcal{G}$ be a* CGS, $s \in \mathrm{St}$ *one of its states,* $\mathrm{P} \subseteq \mathrm{Pth}(s)$ *a set of paths,* $\wp \in \mathrm{Qnt}(\mathrm{V})$ *a quantification prefix over a set* $\mathrm{V} \subseteq \mathrm{Var}$ *of variables, and* $\flat \in \mathrm{Bnd}(\mathrm{V})$ *a binding. Then, player even wins the* TPG $\mathcal{H}(\mathcal{G}, s, \mathrm{P}, \wp, \flat)$ *iff player odd wins the dual* TPG $\mathcal{H}(\mathcal{G}, s, \mathrm{Pth}(s) \setminus \mathrm{P}, \overline{\wp}, \flat)$.

Now, we are going to give the definition of the important concept of *encasement*. Informally, an encasement is a particular subset of paths in a given CGS that "works to encase" an elementary dependence map on strategies, in the sense that it contains all plays obtainable by complete assignments derived from the evaluation of the above mentioned dependence map. In our context, this concept is used to summarize all relevant information needed to verify the elementary satisfiability of a sentence.

Definition 4.22 (*Encasement*). Let $\mathcal{G}$ be a CGS, $s \in \mathrm{St}$ one of its states, $\mathrm{P} \subseteq \mathrm{Pth}(s)$ a set of paths, $\wp \in \mathrm{Qnt}(\mathrm{V})$ a quantification prefix over a set $\mathrm{V} \subseteq \mathrm{Var}$ of variables, and $\flat \in \mathrm{Bnd}(\mathrm{V})$ a binding. Then, P is an *encasement* w.r.t. $\wp$ and $\flat$ if there exists an elementary dependence map $\theta \in \mathrm{EDM}_{\mathrm{Str}(s)}(\wp)$ such that, for all assignments $\chi \in \mathrm{Asg}([[\wp]], s)$, it holds that $\mathrm{play}(\theta(\chi) \circ \zeta_\flat, s) \in \mathrm{P}$.

In the next lemma, we give the second of the two crucial steps in our elementariness proof. In particular, we are able to show a one-to-one relationship between the wining in the dependence-vs-valuation game of player even and the verification of the encasement property of the associated winning set. Moreover, in the case that the latter is a Borelian set, by using Martin's Determinacy Theorem [Martin 1975], we obtain a complete characterization of the winning concept by means of that of encasements.

LEMMA 4.23 (ENCASEMENT CHARACTERIZATION). *Let $\mathcal{G}$ be a* CGS, $s \in \mathrm{St}$ *one of its states,* $\mathrm{P} \subseteq \mathrm{Pth}(s)$ *a set of paths,* $\wp \in \mathrm{Qnt}(\mathrm{V})$ *a quantification prefix over a set* $\mathrm{V} \subseteq \mathrm{Var}$ *of variables, and* $\flat \in \mathrm{Bnd}(\mathrm{V})$ *a binding. Then, the following hold:*

(i) player even wins $\mathcal{H}(\mathcal{G}, s, \mathrm{P}, \wp, \flat)$ iff P is an encasement w.r.t. $\wp$ and $\flat$;
(ii) if player odd wins $\mathcal{H}(\mathcal{G}, s, \mathrm{P}, \wp, \flat)$ then P is not an encasement w.r.t. $\wp$ and $\flat$;
(iii) if P is a Borelian set and it is not an encasement w.r.t. $\wp$ and $\flat$ then player odd wins $\mathcal{H}(\mathcal{G}, s, \mathrm{P}, \wp, \flat)$.

Finally, we have all technical tools useful to prove the elementariness of the satisfiability for SL[1G]. Intuitively, we describe a bidirectional reduction of the problem of interest to the verification of the winning in the dependence-vs-valuation game. The idea behind this construction resides in the strong similarity between the statement of Corollary 4.7 of SL strategy quantification and the definition of the winning condition in a two-player game. Indeed, on one hand, we say that a sentence is satisfiable iff "there exists a dependence map such that, for all all assignments, it holds that ...". On the other hand, we say that player even wins a game iff "there exists an even scheme such that, for all odd schemes, it holds that ...". In particular, for the SL[1G] fragment, we can resolve the gap between these two formulations, by using the concept of elementary quantification.

THEOREM 4.24 (SL[1G] ELEMENTARINESS). *Let $\mathcal{G}$ be a* CGS, φ *an* SL[1G] *formula,* $s \in \mathrm{St}$ *a state, and $\chi \in \mathrm{Asg}(s)$ an s-total assignment with $\mathrm{free}(\varphi) \subseteq \mathrm{dom}(\chi)$. Then, it holds that $\mathcal{G}, \chi, s \models \varphi$ iff $\mathcal{G}, \chi, s \models_\mathrm{E} \varphi$.*

PROOF. The proof proceeds by induction on the structure of the formula. For the sake of succinctness, we only show the most important inductive case of principal subsentences $\phi \in \mathrm{psnt}(\varphi)$, i.e., when ϕ is of the form $\wp\flat\psi$, where $\wp \in \mathrm{Qnt}(\mathrm{V})$ and $\flat \in \mathrm{Bnd}(\mathrm{V})$ are, respectively, a quantification and binding prefix over a set $\mathrm{V} \subseteq \mathrm{Var}$ of variables, and ψ is a variable-closed formula.

[If]. The proof of this direction is practically the same of the one used in Theorem 4.16 of SL[NG] elementary coherence. So, we omit to report it here.

[Only if]. Assume that $\mathcal{G}, \varnothing, s \models \wp\flat\psi$. Then, it is easy to see that, for all elementary dependence maps $\theta \in \mathrm{EDM}_{\mathrm{Str}(s)}(\overline{\wp})$, there is an assignment $\chi \in \mathrm{Asg}(\llbracket\overline{\wp}\rrbracket, s)$ such that $\mathcal{G}, \theta(\chi) \circ \zeta_\flat, s \models \psi$. Indeed, suppose by contradiction that there exists an elementary dependence map $\theta \in \mathrm{EDM}_{\mathrm{Str}(s)}(\overline{\wp})$ such that, for all assignments $\chi \in \mathrm{Asg}(\llbracket\overline{\wp}\rrbracket, s)$, it holds that $\mathcal{G}, \theta(\chi) \circ \zeta_\flat, s \not\models \psi$, i.e., $\mathcal{G}, \theta(\chi) \circ \zeta_\flat, s \models \neg\psi$, and so $\mathcal{G}, \theta(\chi), s \models \flat\neg\psi$. Then, by Corollary 4.7 of SL strategy quantification, we have that $\mathcal{G}, \varnothing, s \models \overline{\wp}\flat\neg\psi$, i.e., $\mathcal{G}, \varnothing, s \models \neg\wp\flat\psi$, and so $\mathcal{G}, \varnothing, s \not\models \wp\flat\psi$, which is impossible.

Now, let $\mathrm{P} \triangleq \{\mathsf{play}(\chi, s) \in \mathrm{Pth}(s) : \chi \in \mathrm{Asg}(\mathrm{Ag}, s) \wedge \mathcal{G}, \chi, s \not\models \psi\}$. Then, it is evident that, for all elementary dependence maps $\theta \in \mathrm{EDM}_{\mathrm{Str}(s)}(\overline{\wp})$, there is an assignment $\chi \in \mathrm{Asg}(\llbracket\overline{\wp}\rrbracket, s)$ such that $\mathsf{play}(\theta(\chi) \circ \zeta_\flat, s) \notin \mathrm{P}$.

At this point, by Definition 4.22 of encasement, it is clear that P is not an encasement w.r.t. $\overline{\wp}$ and $\flat$. Moreover, since ψ describes a regular language, the derived set P is Borelian [Perrin and Pin 2004]. Consequently, by Item iii of Lemma 4.23 of encasement characterization, we have that player odd wins the TPG $\mathcal{H}(\mathcal{G}, s, \mathrm{P}, \overline{\wp}, \flat)$. Thus, by Lemma 4.21 of dependence-vs-valuation duality, player even wins the dual TPG $\mathcal{H}(\mathcal{G}, s, \mathrm{Pth}(s) \setminus \mathrm{P}, \wp, \flat)$. Hence, by Item i of Lemma 4.23, we have that $\mathrm{Pth}(s) \setminus \mathrm{P}$ is an encasement w.r.t. $\wp$ and $\flat$. Finally, again by Definition 4.22, there exists an elementary dependence map $\theta \in \mathrm{EDM}_{\mathrm{Str}(s)}(\wp)$ such that, for all assignments $\chi \in \mathrm{Asg}(\llbracket\wp\rrbracket, s)$, it holds that $\mathsf{play}(\theta(\chi) \circ \zeta_\flat, s) \in \mathrm{Pth}(s) \setminus \mathrm{P}$.

Now, it is immediate to observe that $\mathrm{Pth}(s) \setminus \mathrm{P} = \{\mathsf{play}(\chi, s) \in \mathrm{Pth}(s) : \chi \in \mathrm{Asg}(\mathrm{Ag}, s) \wedge \mathcal{G}, \chi, s \models \psi\}$. So, by the inductive hypothesis, we have that $\mathrm{Pth}(s) \setminus \mathrm{P} = \{\mathsf{play}(\chi, s) \in \mathrm{Pth}(s) : \chi \in \mathrm{Asg}(\mathrm{Ag}, s) \wedge \mathcal{G}, \chi, s \models_\mathrm{E} \psi\}$, from which we derive that there exists an elementary dependence map $\theta \in \mathrm{EDM}_{\mathrm{Str}(s)}(\wp)$ such that, for all assignments $\chi \in \mathrm{Asg}(\llbracket\wp\rrbracket, s)$, it holds that $\mathcal{G}, \theta(\chi) \circ \zeta_\flat, s \models_\mathrm{E} \psi$. Consequently, by Definition 4.11 of SL[NG] elementary semantics, we have that $\mathcal{G}, \varnothing, s \models_\mathrm{E} \wp\flat\psi$. $\square$

As an immediate consequence of the previous theorem, we derive the following fundamental corollary.

COROLLARY 4.25 (SL[1G] ELEMENTARINESS). *Let $\mathcal{G}$ be a* CGS *and φ an* SL[1G] *sentence. Then, $\mathcal{G} \models \varphi$ iff $\mathcal{G} \models_\mathrm{E} \varphi$.*

It is worth to observe that the elementariness property for SL[1G] is a crucial difference w.r.t. SL[BG], which allows us to obtain an elementary decision procedure for the related model-checking problem, as described in the last part of the next section.

5. MODEL-CHECKING PROCEDURES

In this section, we study the model-checking problem for SL and show that, in general, it is non-elementarily decidable, while, in the particular case of SL[1G] sentences, it is just 2ExpTime-complete, as for ATL*. For the algorithmic procedures, we follow an *automata-theoretic approach* [Kupferman et al. 2000], reducing the decision problem for the logics to the emptiness problem of an automaton. In particular, we use a bottom-up technique through which we recursively label each state of the CGS of interest by all principal subsentences of the specification that are satisfied on it, starting from the innermost subsentences and terminating with the sentence under exam. In this way, at a given step of the recursion, since the satisfaction of all subsentences of the given principal sentence has already been determined, we can assume that the matrix of the latter is only composed by Boolean combinations and nesting of goals whose temporal part is simply LTL. The procedure we propose here extends that used for ATL* in [Alur et al. 2002] by means of a richer structure of the automata involved in.

The rest of this section is organized as follows. In Subsection 5.1, we recall the definition of alternating parity tree automata. Then, in Subsection 5.2, we build an automaton accepting a tree encoding of a CGS iff this satisfies the formula of interest, which is used to prove the main result

about SL and SL[NG] model checking. Finally, in Subsection 5.3, we refine the previous result to obtain an elementary decision procedure for SL[1G].

5.1. Alternating tree automata

Nondeterministic tree automata are a generalization to infinite trees of the classical *nondeterministic word automata* on infinite words (see [Thomas 1990], for an introduction). *Alternating tree automata* are a further generalization of nondeterministic tree automata [Muller and Schupp 1987]. Intuitively, on visiting a node of the input tree, while the latter sends exactly one copy of itself to each of the successors of the node, the former can send several own copies to the same successor. Here we use, in particular, *alternating parity tree automata*, which are alternating tree automata along with a *parity acceptance condition* (see [Grädel et al. 2002], for a survey).

We now give the formal definition of alternating tree automata.

Definition 5.1 (*Alternating Tree Automata*). An *alternating tree automaton* (ATA, for short) is a tuple $\mathcal{A} \triangleq \langle \Sigma, \Delta, Q, \delta, q_0, \aleph \rangle$, where Σ, Δ, and Q are, respectively, non-empty finite sets of *input symbols*, *directions*, and *states*, $q_0 \in Q$ is an *initial state*, $\aleph$ is an *acceptance condition* to be defined later, and $\delta : Q \times \Sigma \to B^+(\Delta \times Q)$ is an *alternating transition function* that maps each pair of states and input symbols to a positive Boolean combination on the set of propositions of the form $(d, q) \in \Delta \times Q$, a.k.a. *moves*.

On one side, a *nondeterministic tree automaton* (NTA, for short) is a special case of ATA in which each conjunction in the transition function δ has exactly one move (d, q) associated with each direction d. This means that, for all states $q \in Q$ and symbols $\sigma \in \Sigma$, we have that $\delta(q, \sigma)$ is equivalent to a Boolean formula of the form $\bigvee_i \bigwedge_{d \in \Delta}(d, q_{i,d})$. On the other side, a *universal tree automaton* (UTA, for short) is a special case of ATA in which all the Boolean combinations that appear in δ are conjunctions of moves. Thus, we have that $\delta(q, \sigma) = \bigwedge_i (d_i, q_i)$, for all states $q \in Q$ and symbols $\sigma \in \Sigma$.

The semantics of the ATAs is given through the following concept of run.

Definition 5.2 (ATA *Run*). A *run* of an ATA $\mathcal{A} = \langle \Sigma, \Delta, Q, \delta, q_0, \aleph \rangle$ on a Σ-labeled Δ-tree $\mathcal{T} = \langle T, v \rangle$ is a $(\Delta \times Q)$-tree R such that, for all nodes $x \in R$, where $x = \prod_{i=1}^n (d_i, q_i)$ and $y \triangleq \prod_{i=1}^n d_i$ with $n \in [0, \omega[$, it holds that *(i)* $y \in T$ and *(ii)*, there is a set of moves $S \subseteq \Delta \times Q$ with $S \models \delta(q_n, v(y))$ such that $x \cdot (d, q) \in R$, for all $(d, q) \in S$.

In the following, we consider ATAs along with the *parity acceptance condition* (APT, for short) $\aleph \triangleq (F_1, \ldots, F_k) \in (2^Q)^+$ with $F_1 \subseteq \ldots \subseteq F_k = Q$ (see [Kupferman et al. 2000], for more). The number k of sets in the tuple $\aleph$ is called the *index* of the automaton. We also consider ATAs with the *co-Büchi acceptance condition* (ACT, for short) that is the special parity condition with index 2.

Let R be a run of an ATA $\mathcal{A}$ on a tree $\mathcal{T}$ and w one of its branches. Then, by $\inf(w) \triangleq \{q \in Q : |\{i \in \mathbb{N} : \exists d \in \Delta.(w)_i = (d, q)\}| = \omega\}$ we denote the set of states that occur infinitely often as the second component of the letters along the branch w. Moreover, we say that w satisfies the parity acceptance condition $\aleph = (F_1, \ldots, F_k)$ if the least index $i \in [1, k]$ for which $\inf(w) \cap F_i \neq \emptyset$ is even.

At this point, we can define the concept of language accepted by an ATA.

Definition 5.3 (ATA *Acceptance*). An ATA $\mathcal{A} = \langle \Sigma, \Delta, Q, \delta, q_0, \aleph \rangle$ *accepts* a Σ-labeled Δ-tree $\mathcal{T}$ iff is there exists a run R of $\mathcal{A}$ on $\mathcal{T}$ such that all its infinite branches satisfy the acceptance condition $\aleph$.

By $L(\mathcal{A})$ we denote the language accepted by the ATA $\mathcal{A}$, i.e., the set of trees $\mathcal{T}$ accepted by $\mathcal{A}$. Moreover, $\mathcal{A}$ is said to be *empty* if $L(\mathcal{A}) = \emptyset$. The *emptiness problem* for $\mathcal{A}$ is to decide whether $L(\mathcal{A}) = \emptyset$.

We finally show a simple but useful result about the APT direction projection. To do this, we first need to introduce an extra notation. Let $f \in B(P)$ be a Boolean formula on a set of propositions

P. Then, by $f[p/q \mid p \in \mathrm{P}']$ we denote the formula in which all occurrences of the propositions $p \in \mathrm{P}' \subseteq \mathrm{P}$ in f are replaced by the proposition q belonging to a possibly different set.

THEOREM 5.4 (APT DIRECTION PROJECTION). *Let* $\mathcal{A} \triangleq \langle \Sigma \times \Delta, \Delta, \mathrm{Q}, \delta, q_0, \aleph \rangle$ *be an* APT *over a set of* m *directions with* n *states and index* k. *Moreover, let* $d_0 \in \Delta$ *be a distinguished direction. Then, there exists an* NPT $\mathcal{N}^{d_0} \triangleq \langle \Sigma, \Delta, \mathrm{Q}', \delta', q_0', \aleph' \rangle$ *with* $m \cdot 2^{\mathrm{O}(k \cdot n \cdot \log n)}$ *states and index* $\mathrm{O}(k \cdot n \cdot \log n)$ *such that, for all* Σ-*labeled* Δ-*tree* $\mathcal{T} \triangleq \langle \mathrm{T}, \mathsf{v} \rangle$, *it holds that* $\mathcal{T} \in \mathrm{L}(\mathcal{N}^{d_0})$ *iff* $\mathcal{T}' \in \mathrm{L}(\mathcal{A})$, *where* $\mathcal{T}'$ *is the* $(\Sigma \times \Delta)$-*labeled* Δ-*tree* $\langle \mathrm{T}, \mathsf{v}' \rangle$ *such that* $\mathsf{v}'(t) \triangleq (\mathsf{v}(t), \mathsf{lst}(d_0 \cdot t))$, *for all* $t \in \mathrm{T}$.

PROOF. As first step, we use the well-known nondeterminization procedure for APTs [Muller and Schupp 1995] in order to transform the APT $\mathcal{A}$ into an equivalent NPT $\mathcal{N} = \langle \Sigma \times \Delta, \Delta, \mathrm{Q}'', \delta'', q_0'', \aleph'' \rangle$ with $2^{\mathrm{O}(k \cdot n \cdot \log n)}$ states and index $k' = \mathrm{O}(k \cdot n \cdot \log n)$. Then, we transform the latter into the new NPT $\mathcal{N}^{d_0} \triangleq \langle \Sigma, \Delta, \mathrm{Q}', \delta', q_0', \aleph' \rangle$ with $m \cdot 2^{\mathrm{O}(k \cdot n \cdot \log n)}$ states and same index k', where $\mathrm{Q}' \triangleq \mathrm{Q}'' \times \Delta$, $q_0' \triangleq (q_0'', d_0)$, $\aleph' \triangleq (\mathrm{F}_1 \times \Delta, \ldots, \mathrm{F}_{k'} \times \Delta)$ with $\aleph'' \triangleq (\mathrm{F}_1, \ldots, \mathrm{F}_{k'})$, and $\delta'((q,d), \sigma) \triangleq \delta''(q, (\sigma, d))[(d', q')/(d', (q', d')) \mid (d', q') \in \Delta \times \mathrm{Q}'']$, for all $(q, d) \in \mathrm{Q}'$ and $\sigma \in \Sigma$. Now, it easy to see that $\mathcal{N}^{d_0}$ satisfies the declared statement. □

5.2. SL Model Checking

A first step towards our construction of an algorithmic procedure for the solution of the SL model-checking problem is to define, for each possible formula φ, an alternating parity tree automaton $\mathcal{A}_\varphi^{\mathcal{G}}$ that recognizes a tree encoding $\mathcal{T}$ of a CGS $\mathcal{G}$, containing the information on an assignment χ on the free variables/agents of φ, iff $\mathcal{G}$ is a model of φ under χ. The high-level idea at the base of this construction is an evolution and merging of those behind the translations of QPTL and LTL, respectively, into nondeterministic [Sistla et al. 1987] and alternating [Muller et al. 1988] Büchi automata.

To proceed with the formal description of the model-checking procedure, we have to introduce a concept of encoding for the assignments of a CGS.

Definition 5.5 (*Assignment-State Encoding*). Let $\mathcal{G}$ be a CGS, $s \in \mathrm{St}_{\mathcal{G}}$ one of its states, and $\chi \in \mathrm{Asg}_{\mathcal{G}}(\mathrm{V}, s)$ an assignment defined on the set $\mathrm{V} \subseteq \mathrm{Var} \cup \mathrm{Ag}$. Then, a $(\mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathrm{V}) \times \mathrm{St}_{\mathcal{G}})$-labeled $\mathrm{St}_{\mathcal{G}}$-tree $\mathcal{T} \triangleq \langle \mathrm{T}, \mathsf{u} \rangle$, where $\mathrm{T} \triangleq \{\rho_{\geq 1} : \rho \in \mathrm{Trk}_{\mathcal{G}}(s)\}$, is an *assignment-state encoding* for χ if it holds that $\mathsf{u}(t) \triangleq (\widehat{\chi}(s \cdot t), \mathsf{lst}(s \cdot t))$, for all $t \in \mathrm{T}$.

Observe that there is a unique assignment-state encoding for each given assignment.

In the next lemma, we prove the existence of an APT for each CGS and SL formula that is able to recognize all the assignment-state encodings of an a priori given assignment, made the assumption that the formula is satisfied on the CGS under this assignment.

LEMMA 5.6 (SL FORMULA AUTOMATON). *Let* $\mathcal{G}$ *be a* CGS *and* φ *an* SL *formula. Then, there exists an* APT $\mathcal{A}_\varphi^{\mathcal{G}} \triangleq \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathrm{free}(\varphi)) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_\varphi, \delta_\varphi, q_{0\varphi}, \aleph_\varphi \rangle$ *such that, for all states* $s \in \mathrm{St}_{\mathcal{G}}$ *and assignments* $\chi \in \mathrm{Asg}_{\mathcal{G}}(\mathrm{free}(\varphi), s)$, *it holds that* $\mathcal{G}, \chi, s \models \varphi$ *iff* $\mathcal{T} \in \mathrm{L}(\mathcal{A}_\varphi^{\mathcal{G}})$, *where* $\mathcal{T}$ *is the assignment-state encoding for* χ.

PROOF. The construction of the APT $\mathcal{A}_\varphi^{\mathcal{G}}$ is done recursively on the structure of the formula φ, which w.l.o.g. is supposed to be in *enf*, by using a variation of the transformation, via alternating tree automata, of the S1S and SkS logics into nondeterministic Büchi word and tree automata recognizing all models of the formula of interest [Büchi 1962; Rabin 1969].

The detailed construction of $\mathcal{A}_\varphi^{\mathcal{G}}$, by a case analysis on φ, follows.

— If $\varphi \in \mathrm{AP}$, the automaton has to verify if the atomic proposition is locally satisfied or not. To do this, we set $\mathcal{A}_\varphi^{\mathcal{G}} \triangleq \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\emptyset) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \{\varphi\}, \delta_\varphi, \varphi, (\{\varphi\}) \rangle$, where $\delta_\varphi(\varphi, (\mathsf{v}, s)) \triangleq \mathsf{t}$, if $\varphi \in \lambda_{\mathcal{G}}(s)$, and $\delta_\varphi(\varphi, (\mathsf{v}, s)) \triangleq \mathsf{f}$, otherwise. Intuitively, $\mathcal{A}_\varphi^{\mathcal{G}}$ only verifies that the state s in the labeling of the root of the assignment-state encoding of the empty assignment $\varnothing$ satisfies φ.

— The boolean case $\varphi = \neg\varphi'$ is treated in the classical way, by simply dualizing the automaton $\mathcal{A}_{\varphi'}^{\mathcal{G}} = \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathrm{free}(\varphi')) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\varphi'}, \delta_{\varphi'}, q_{0\varphi'}, \aleph_{\varphi'} \rangle$ [Muller and Schupp 1987].

— The boolean cases $\varphi = \varphi_1 \mathsf{Op}\, \varphi_2$, with $\mathsf{Op} \in \{\wedge, \vee\}$, are treated in a way that is similar to the classical one, by simply merging the two automata $\mathcal{A}_{\varphi_1}^{\mathcal{G}} = \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathrm{free}(\varphi_1)) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}},$ $\mathrm{Q}_{\varphi_1}, \delta_{\varphi_1}, q_{0\varphi_1}, \aleph_{\varphi_1} \rangle$ and $\mathcal{A}_{\varphi_2}^{\mathcal{G}} = \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathrm{free}(\varphi_2)) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\varphi_2}, \delta_{\varphi_2}, q_{0\varphi_2}, \aleph_{\varphi_2} \rangle$ into the automaton $\mathcal{A}_{\varphi}^{\mathcal{G}} \triangleq \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathrm{free}(\varphi)) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\varphi}, \delta_{\varphi}, q_{0\varphi}, \aleph_{\varphi} \rangle$, where the following hold:

  — $\mathrm{Q}_{\varphi} \triangleq \{q_{0\varphi}\} \cup \mathrm{Q}_{\varphi_1} \cup \mathrm{Q}_{\varphi_2}$, with $q_{0\varphi} \notin \mathrm{Q}_{\varphi_1} \cup \mathrm{Q}_{\varphi_2}$;
  — $\delta_{\varphi}(q_{0\varphi}, (\mathsf{v}, s)) \triangleq \delta_{\varphi_1}(q_{0\varphi_1}, (\mathsf{v}_{\upharpoonright \mathrm{free}(\varphi_1)}, s))\ \mathsf{Op}\ \delta_{\varphi_2}(q_{0\varphi_2}, (\mathsf{v}_{\upharpoonright \mathrm{free}(\varphi_2)}, s))$, for all $(\mathsf{v}, s) \in \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathrm{free}(\varphi)) \times \mathrm{St}_{\mathcal{G}}$;
  — $\delta_{\varphi}(q, (\mathsf{v}, s)) \triangleq \delta_{\varphi_1}(q, (\mathsf{v}_{\upharpoonright \mathrm{free}(\varphi_1)}, s))$, if $q \in \mathrm{Q}_{\varphi_1}$, and $\delta_{\varphi}(q, (\mathsf{v}, s)) \triangleq \delta_{\varphi_2}(q, (\mathsf{v}_{\upharpoonright \mathrm{free}(\varphi_2)}, s))$, otherwise, for all $q \in \mathrm{Q}_{\varphi_1} \cup \mathrm{Q}_{\varphi_2}$ and $(\mathsf{v}, s) \in \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathrm{free}(\varphi)) \times \mathrm{St}_{\mathcal{G}}$;
  — $\aleph_{\varphi} \triangleq (\mathrm{F}_{1\varphi}, \ldots, \mathrm{F}_{k\varphi})$, where *(i)* $\aleph_{\varphi_1} \triangleq (\mathrm{F}_{1\varphi_1}, \ldots, \mathrm{F}_{k_1\varphi_1})$ and $\aleph_{\varphi_2} \triangleq (\mathrm{F}_{1\varphi_2}, \ldots, \mathrm{F}_{k_2\varphi_2})$, *(ii)* $h = \min\{k_1, k_2\}$ and $k = \max\{k_1, k_2\}$, *(iii)* $\mathrm{F}_{i\varphi} \triangleq \mathrm{F}_{i\varphi_1} \cup \mathrm{F}_{i\varphi_2}$, for $i \in [1, h]$, *(iv)* $\mathrm{F}_{i\varphi} \triangleq \mathrm{F}_{i\varphi_j}$, for $i \in [h+1, k-1]$ with $k_j = k$, and *(v)* $\mathrm{F}_{k\varphi} \triangleq \mathrm{Q}_{\varphi}$.

— The case $\varphi = \mathsf{X}\,\varphi'$ is solved by running the automaton $\mathcal{A}_{\varphi'}^{\mathcal{G}} = \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathrm{free}(\varphi')) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\varphi'},$ $\delta_{\varphi'}, q_{0\varphi'}, \aleph_{\varphi'} \rangle$ on the successor node of the root of the assignment-state encoding in the direction individuated by the assignment itself. To do this, we use the automaton $\mathcal{A}_{\varphi}^{\mathcal{G}} \triangleq \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathrm{free}(\varphi)) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\varphi}, \delta_{\varphi}, q_{0\varphi}, \aleph_{\varphi} \rangle$, where the following hold:

  — $\mathrm{Q}_{\varphi} \triangleq \{q_{0\varphi}\} \cup \mathrm{Q}_{\varphi'}$, with $q_{0\varphi} \notin \mathrm{Q}_{\varphi'}$;
  — $\delta_{\varphi}(q_{0\varphi}, (\mathsf{v}, s)) \triangleq (\tau_{\mathcal{G}}(s, \mathsf{v}_{\upharpoonright \mathrm{Ag}}), q_{0\varphi'})$, for all $(\mathsf{v}, s) \in \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathrm{free}(\varphi)) \times \mathrm{St}_{\mathcal{G}}$;
  — $\delta_{\varphi}(q, (\mathsf{v}, s)) \triangleq \delta_{\varphi'}(q, (\mathsf{v}_{\upharpoonright \mathrm{free}(\varphi')}, s))$, for all $q \in \mathrm{Q}_{\varphi'}$ and $(\mathsf{v}, s) \in \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathrm{free}(\varphi)) \times \mathrm{St}_{\mathcal{G}}$;
  — $\aleph_{\varphi} \triangleq (\mathrm{F}_{1\varphi'}, \ldots, \mathrm{F}_{k\varphi'} \cup \{q_{0\varphi}\})$, where $\aleph_{\varphi'} \triangleq (\mathrm{F}_{1\varphi'}, \ldots, \mathrm{F}_{k\varphi'})$.

— To handle the case $\varphi = \varphi_1 \mathsf{U}\, \varphi_2$, we use the automaton $\mathcal{A}_{\varphi}^{\mathcal{G}} \triangleq \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathrm{free}(\varphi)) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\varphi}, \delta_{\varphi}, q_{0\varphi}, \aleph_{\varphi} \rangle$ that verifies the truth of the until operator using its one-step unfolding equivalence $\varphi_1 \mathsf{U}\, \varphi_2 \equiv \varphi_2 \vee \varphi_1 \wedge \mathsf{X}\,\varphi_1 \mathsf{U}\, \varphi_2$, by appropriately running the two automata $\mathcal{A}_{\varphi_1}^{\mathcal{G}} = \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathrm{free}(\varphi_1)) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\varphi_1}, \delta_{\varphi_1}, q_{0\varphi_1}, \aleph_{\varphi_1} \rangle$ and $\mathcal{A}_{\varphi_2}^{\mathcal{G}} = \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathrm{free}(\varphi_2)) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\varphi_2}, \delta_{\varphi_2}, q_{0\varphi_2}, \aleph_{\varphi_2} \rangle$ for the inner formulas φ_1 and φ_2. The definitions of $\mathcal{A}_{\varphi}^{\mathcal{G}}$ components follows:

  — $\mathrm{Q}_{\varphi} \triangleq \{q_{0\varphi}\} \cup \mathrm{Q}_{\varphi_1} \cup \mathrm{Q}_{\varphi_2}$, with $q_{0\varphi} \notin \mathrm{Q}_{\varphi_1} \cup \mathrm{Q}_{\varphi_2}$;
  — $\delta_{\varphi}(q_{0\varphi}, (\mathsf{v}, s)) \triangleq \delta_{\varphi_2}(q_{0\varphi_2}, (\mathsf{v}_{\upharpoonright \mathrm{free}(\varphi_2)}, s)) \vee \delta_{\varphi_1}(q_{0\varphi_1}, (\mathsf{v}_{\upharpoonright \mathrm{free}(\varphi_1)}, s)) \wedge (\tau_{\mathcal{G}}(s, \mathsf{v}_{\upharpoonright \mathrm{Ag}}), q_{0\varphi})$, for all $(\mathsf{v}, s) \in \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathrm{free}(\varphi)) \times \mathrm{St}_{\mathcal{G}}$;
  — $\delta_{\varphi}(q, (\mathsf{v}, s)) \triangleq \delta_{\varphi_1}(q, (\mathsf{v}_{\upharpoonright \mathrm{free}(\varphi_1)}, s))$, if $q \in \mathrm{Q}_{\varphi_1}$, and $\delta_{\varphi}(q, (\mathsf{v}, s)) \triangleq \delta_{\varphi_2}(q, (\mathsf{v}_{\upharpoonright \mathrm{free}(\varphi_2)}, s))$, otherwise, for all $q \in \mathrm{Q}_{\varphi_1} \cup \mathrm{Q}_{\varphi_2}$ and $(\mathsf{v}, s) \in \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathrm{free}(\varphi)) \times \mathrm{St}_{\mathcal{G}}$;
  — $\aleph_{\varphi} \triangleq (\mathrm{F}_{1\varphi}, \ldots, \mathrm{F}_{k\varphi})$, where *(i)* $\aleph_{\varphi_1} \triangleq (\mathrm{F}_{1\varphi_1}, \ldots, \mathrm{F}_{k_1\varphi_1})$ and $\aleph_{\varphi_2} \triangleq (\mathrm{F}_{1\varphi_2}, \ldots, \mathrm{F}_{k_2\varphi_2})$, *(ii)* $h = \min\{k_1, k_2\}$ and $k = \max\{k_1, k_2\}$, *(iii)* $\mathrm{F}_{i\varphi} \triangleq \{q_{0\varphi}\} \cup \mathrm{F}_{i\varphi_1} \cup \mathrm{F}_{i\varphi_2}$, for $i \in [1, h]$, *(iv)* $\mathrm{F}_{i\varphi} \triangleq \{q_{0\varphi}\} \cup \mathrm{F}_{i\varphi_j}$, for $i \in [h+1, k-1]$ with $k_j = k$, and *(v)* $\mathrm{F}_{k\varphi} \triangleq \mathrm{Q}_{\varphi}$.
  It is important to observe that the initial state $q_{0\varphi}$ is included in all sets of the parity acceptance condition, in particular in $\mathrm{F}_{1\varphi}$, in order to avoid its regeneration for an infinite number of times.

— To handle the case $\varphi = \varphi_1 \mathsf{R}\, \varphi_2$, we use the automaton $\mathcal{A}_{\varphi}^{\mathcal{G}} \triangleq \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathrm{free}(\varphi)) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\varphi}, \delta_{\varphi}, q_{0\varphi}, \aleph_{\varphi} \rangle$ that verifies the truth of the release operator using its one-step unfolding equivalence $\varphi_1 \mathsf{R}\, \varphi_2 \equiv \varphi_2 \wedge (\varphi_1 \vee \mathsf{X}\,\varphi_1 \mathsf{R}\, \varphi_2)$, by appropriately running the two automata $\mathcal{A}_{\varphi_1}^{\mathcal{G}} = \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathrm{free}(\varphi_1)) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\varphi_1}, \delta_{\varphi_1}, q_{0\varphi_1}, \aleph_{\varphi_1} \rangle$ and $\mathcal{A}_{\varphi_2}^{\mathcal{G}} = \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathrm{free}(\varphi_2)) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\varphi_2}, \delta_{\varphi_2}, q_{0\varphi_2}, \aleph_{\varphi_2} \rangle$ for the inner formulas φ_1 and φ_2. The definitions of $\mathcal{A}_{\varphi}^{\mathcal{G}}$ components follows:

  — $\mathrm{Q}_{\varphi} \triangleq \{q_{0\varphi}\} \cup \mathrm{Q}_{\varphi_1} \cup \mathrm{Q}_{\varphi_2}$, with $q_{0\varphi} \notin \mathrm{Q}_{\varphi_1} \cup \mathrm{Q}_{\varphi_2}$;

— $\delta_\varphi(q_{0\varphi}, (\mathsf{v}, s)) \triangleq \delta_{\varphi_2}(q_{0\varphi_2}, (\mathsf{v}_{\restriction \mathsf{free}(\varphi_2)}, s)) \wedge (\delta_{\varphi_1}(q_{0\varphi_1}, (\mathsf{v}_{\restriction \mathsf{free}(\varphi_1)}, s)) \vee (\tau_\mathcal{G}(s, \mathsf{v}_{\restriction \mathrm{Ag}}), q_{0\varphi}))$, for all $(\mathsf{v}, s) \in \mathrm{Val}_{\mathrm{Ac}_\mathcal{G}}(\mathsf{free}(\varphi)) \times \mathrm{St}_\mathcal{G}$;

— $\delta_\varphi(q, (\mathsf{v}, s)) \triangleq \delta_{\varphi_1}(q, (\mathsf{v}_{\restriction \mathsf{free}(\varphi_1)}, s))$, if $q \in \mathrm{Q}_{\varphi_1}$, and $\delta_\varphi(q, (\mathsf{v}, s)) \triangleq \delta_{\varphi_2}(q, (\mathsf{v}_{\restriction \mathsf{free}(\varphi_2)}, s))$, otherwise, for all $q \in \mathrm{Q}_{\varphi_1} \cup \mathrm{Q}_{\varphi_2}$ and $(\mathsf{v}, s) \in \mathrm{Val}_{\mathrm{Ac}_\mathcal{G}}(\mathsf{free}(\varphi)) \times \mathrm{St}_\mathcal{G}$;

— $\aleph_\varphi \triangleq (\mathrm{F}_{1\varphi}, \ldots, \mathrm{F}_{k\varphi})$, where *(i)* $\aleph_{\varphi_1} \triangleq (\mathrm{F}_{1\varphi_1}, \ldots, \mathrm{F}_{k_1\varphi_1})$ and $\aleph_{\varphi_2} \triangleq (\mathrm{F}_{1\varphi_2}, \ldots, \mathrm{F}_{k_2\varphi_2})$, *(ii)* $h = \min\{k_1, k_2\}$ and $k = \max\{k_1, k_2\}$, *(iii)* $\mathrm{F}_{1\varphi} \triangleq \mathrm{F}_{1\varphi_1} \cup \mathrm{F}_{1\varphi_2}$, *(iv)* $\mathrm{F}_{i\varphi} \triangleq \{q_{0\varphi}\} \cup \mathrm{F}_{i\varphi_1} \cup \mathrm{F}_{i\varphi_2}$, for $i \in [2, h]$, *(iv)* $\mathrm{F}_{i\varphi} \triangleq \{q_{0\varphi}\} \cup \mathrm{F}_{i\varphi_j}$, for $i \in [h+1, k-1]$ with $k_j = k$, and *(v)* $\mathrm{F}_{k\varphi} \triangleq \mathrm{Q}_\varphi$.

It is important to observe that, differently from the case of the until operator, the initial state $q_{0\varphi}$ is included in all sets of the parity acceptance condition but $\mathrm{F}_{1\varphi}$, in order to allow its regeneration for an infinite number of time.

— The case $\varphi = (a, x)\varphi'$ is solved by simply transforming the transition function of the automaton $\mathcal{A}_{\varphi'}^\mathcal{G} = \langle \mathrm{Val}_{\mathrm{Ac}_\mathcal{G}}(\mathsf{free}(\varphi')) \times \mathrm{St}_\mathcal{G}, \mathrm{St}_\mathcal{G}, \mathrm{Q}_{\varphi'}, \delta_{\varphi'}, q_{0\varphi'}, \aleph_{\varphi'} \rangle$, by setting the value of the valuations in input w.r.t. the agent a to the value of the same valuation w.r.t. the variable x. The definitions of the transition function for $\mathcal{A}_\varphi^\mathcal{G} \triangleq \langle \mathrm{Val}_{\mathrm{Ac}_\mathcal{G}}(\mathsf{free}(\varphi)) \times \mathrm{St}_\mathcal{G}, \mathrm{St}_\mathcal{G}, \mathrm{Q}_{\varphi'}, \delta_\varphi, q_{0\varphi'}, \aleph_{\varphi'} \rangle$ follows: $\delta_\varphi(q, (\mathsf{v}, s)) \triangleq \delta_{\varphi'}(q, (\mathsf{v}', s))$, where $\mathsf{v}' = \mathsf{v}[a \mapsto \mathsf{v}(x)]_{\restriction \mathsf{free}(\varphi')}$, if $a \in \mathsf{free}(\varphi')$, and $\mathsf{v}' = \mathsf{v}$, otherwise, for all $q \in \mathrm{Q}_{\varphi'}$ and $(\mathsf{v}, s) \in \mathrm{Val}_{\mathrm{Ac}_\mathcal{G}}(\mathsf{free}(\varphi)) \times \mathrm{St}_\mathcal{G}$.

— To handle the case $\varphi = \langle\langle x \rangle\rangle \varphi'$, assuming that $x \in \mathsf{free}(\varphi')$, we use the operation of existential projection for nondeterministic tree automata. To do this, we have first to nondeterminize the APT $\mathcal{A}_{\varphi'}^\mathcal{G}$, by applying the classic transformation [Muller and Schupp 1995]. In this way, we obtain an equivalent NPT $\mathcal{N}_{\varphi'}^\mathcal{G} = \langle \mathrm{Val}_{\mathrm{Ac}_\mathcal{G}}(\mathsf{free}(\varphi')) \times \mathrm{St}_\mathcal{G}, \mathrm{St}_\mathcal{G}, \mathrm{Q}_{\varphi'}, \delta_{\varphi'}, q_{0\varphi'}, \aleph_{\varphi'} \rangle$. Now, we make the projection, by defining the new NPT $\mathcal{A}_\varphi^\mathcal{G} \triangleq \langle \mathrm{Val}_{\mathrm{Ac}_\mathcal{G}}(\mathsf{free}(\varphi)) \times \mathrm{St}_\mathcal{G}, \mathrm{St}_\mathcal{G}, \mathrm{Q}_{\varphi'}, \delta_\varphi, q_{0\varphi'}, \aleph_{\varphi'} \rangle$ where $\delta_\varphi(q, (\mathsf{v}, s)) \triangleq \bigvee_{c \in \mathrm{Ac}_\mathcal{G}} \delta_{\varphi'}(q, (\mathsf{v}[x \mapsto c], s))$, for all $q \in \mathrm{Q}_{\varphi'}$ and $(\mathsf{v}, s) \in \mathrm{Val}_{\mathrm{Ac}_\mathcal{G}}(\mathsf{free}(\varphi)) \times \mathrm{St}_\mathcal{G}$.

At this point, it only remains to prove that, for all states $s \in \mathrm{St}_\mathcal{G}$ and assignments $\chi \in \mathrm{Asg}_\mathcal{G}(\mathsf{free}(\varphi), s)$, it holds that $\mathcal{G}, \chi, s \models \varphi$ iff $\mathcal{T} \in \mathrm{L}(\mathcal{A}_\varphi^\mathcal{G})$, where $\mathcal{T}$ is the assignment-state encoding for χ. The proof can be developed by a simple induction on the structure of the formula φ and is left to the reader as a simple exercise. $\square$

We now have the tools to describe the recursive model-checking procedure on nested subsentences for SL and its fragments under the general semantics.

To proceed, we have first to prove the following theorem that represents the core of our automata-theoretic approach.

THEOREM 5.7 (SL SENTENCE AUTOMATON). *Let $\mathcal{G}$ be a CGS, $s \in \mathrm{St}_\mathcal{G}$ one of its states, and φ an SL sentence. Then, there exists an NPT $\mathcal{N}_\varphi^{\mathcal{G},s}$ such that $\mathcal{G}, \varnothing, s \models \varphi$ iff $\mathrm{L}(\mathcal{N}_\varphi^{\mathcal{G},s}) \neq \emptyset$.*

PROOF. To construct the NPT $\mathcal{N}_\varphi^{\mathcal{G},s}$ we apply Theorem 5.4 of APT direction projection with distinguished direction s to the APT $\mathcal{A}_\varphi^\mathcal{G}$ derived by Lemma 5.6 of SL formula automaton. In this way, we can ensure that the state labeling of nodes of the assignment-state encoding is coherent with the node itself. Observe that, since φ is a sentence, we have that $\mathsf{free}(\varphi) = \emptyset$, and so, the unique assignment-state encoding of interest is that related to the empty assignment $\varnothing$.

[Only if]. Suppose that $\mathcal{G}, \varnothing, s \models \varphi$. Then, by Lemma 5.6, we have that $\mathcal{T} \in \mathrm{L}(\mathcal{A}_\varphi^\mathcal{G})$, where $\mathcal{T}$ is the elementary dependence-state encoding for $\varnothing$. Hence, by Theorem 5.4, it holds that $\mathrm{L}(\mathcal{N}_\varphi^{\mathcal{G},s}) \neq \emptyset$.

[If]. Suppose that $\mathrm{L}(\mathcal{N}_\varphi^{\mathcal{G},s}) \neq \emptyset$. Then, by Theorem 5.4, there exists an $(\{\varnothing\} \times \mathrm{St}_\mathcal{G})$-labeled $\mathrm{St}_\mathcal{G}$-tree $\mathcal{T}$ such that $\mathcal{T} \in \mathrm{L}(\mathcal{A}_\varphi^\mathcal{G})$. Now, it is immediate to see that $\mathcal{T}$ is the assignment-state encoding for $\varnothing$. Hence, by Lemma 5.6, we have that $\mathcal{G}, \varnothing, s \models \varphi$. $\square$

Before continuing, we define the length $\mathsf{lng}(\varphi)$ of an SL formula φ as the number $|\mathsf{sub}(\varphi)|$ of its subformulas. We also introduce a generalization of the Knuth's double arrow notation in order to represents a tower of exponentials: $a \uparrow\uparrow_b 0 \triangleq b$ and $a \uparrow\uparrow_b (c+1) \triangleq a^{a\uparrow\uparrow_b c}$, for all $a, b, c \in \mathbb{N}$.

At this point, we prove the main theorem about the non-elementary complexity of SL model-checking problem.

THEOREM 5.8 (SL MODEL CHECKING). *The model-checking problem for* SL *is* PTIME-COMPLETE *w.r.t. the size of the model and* NONELEMENTARYTIME *w.r.t. the size of the specification.*

PROOF. By Theorem 5.7 of SL sentence automaton, to verify that $\mathcal{G}, \varnothing, s \models \varphi$, we simply calculate the emptiness of the NPT $\mathcal{N}_\varphi^{\mathcal{G},s}$ having $|\mathrm{St}_\mathcal{G}| \cdot (2 \uparrow\uparrow_m m)$ states and index $2 \uparrow\uparrow_m m$, where $m = \mathsf{O}(\mathsf{lng}(\varphi) \cdot \log \mathsf{lng}(\varphi))$. It is well-known that the emptiness problem for such a kind of automaton with n states and index h is solvable in time $\mathsf{O}(n^h)$ [Kupferman and Vardi 1998]. Thus, we get that the time complexity of checking whether $\mathcal{G}, \varnothing, s \models \varphi$ is $|\mathrm{St}_\mathcal{G}|^{2\uparrow\uparrow_m m}$. Hence, the membership of the model-checking problem for SL in PTIME w.r.t. the size of the model and NONELEMENTARYTIME w.r.t. the size of the specification directly follows. Finally, by getting the relative lower bound on the model from the same problem for ATL* [Alur et al. 2002], the thesis is proved. □

Finally, we show a refinement of the previous result, when we consider sentences of the SL[NG] fragment.

THEOREM 5.9 (SL[NG] MODEL CHECKING). *The model-checking problem for* SL[NG] *is* PTIME-COMPLETE *w.r.t. the size of the model and* $(k+1)$-EXPTIME *w.r.t. the maximum alternation* k *of the specification.*

PROOF. By Theorem 5.7 of SL sentence automaton, to verify that $\mathcal{G}, \varnothing, s \models \wp\psi$, where $\wp\psi$ is an SL[NG] principal sentence without proper subsentences, we can simply calculate the emptiness of the NPT $\mathcal{N}_{\wp\psi}^{\mathcal{G},s}$ having $|\mathrm{St}_\mathcal{G}| \cdot (2 \uparrow\uparrow_m k)$ states and index $2 \uparrow\uparrow_m k$, where $m = \mathsf{O}(\mathsf{lng}(\psi) \cdot \log \mathsf{lng}(\psi))$ and $k = \mathsf{alt}(\wp\psi)$. Thus, we get that the time complexity of checking whether $\mathcal{G}, \varnothing, s \models \wp\psi$ is $|\mathrm{St}_\mathcal{G}|^{2\uparrow\uparrow_m k}$. At this point, since we have to do this verification for each possible state $s \in \mathrm{St}_\mathcal{G}$ and principal subsentence $\wp\psi \in \mathsf{psnt}(\varphi)$ of the whole SL[NG] specification φ, we derive that the bottom-up model-checking procedure requires time $|\mathrm{St}_\mathcal{G}|^{2\uparrow\uparrow_{\mathsf{lng}(\varphi)} k}$, where $k = \max\{\mathsf{alt}(\wp\psi)$: $\wp\psi \in \mathsf{psnt}(\varphi)\}$. Hence, the membership of the model-checking problem for SL in PTIME w.r.t. the size of the model and $(k+1)$-EXPTIME w.r.t. the maximum alternation k of the specification directly follows. Finally, by getting the relative lower bound on the model from the same problem for ATL* [Alur et al. 2002], the thesis is proved. □

5.3. SL[1G] Model Checking

We now show how the concept of elementariness of dependence maps over strategies can be used to enormously reduce the complexity of the model-checking procedure for the SL[1G] fragment. The idea behind our approach is to avoid the use of projections used to handle the strategy quantifications, by reducing them to action quantifications, which can be managed locally on each state of the model without a tower of exponential blow-ups.

To start with the description of the ad-hoc procedure for SL[1G], we first prove the existence of a UCT for each CGS and SL[1G] goal $\flat\psi$ that recognizes all the assignment-state encodings of an a priori given assignment, made the assumption that the goal is satisfied on the CGS under this assignment.

LEMMA 5.10 (SL[1G] GOAL AUTOMATON). *Let* $\mathcal{G}$ *be a* CGS *and* $\flat\psi$ *an* SL[1G] *goal without principal subsentences. Then, there exists an* UCT $\mathcal{U}_{\flat\psi}^{\mathcal{G}} \triangleq \langle \mathrm{Val}_{\mathrm{Ac}_\mathcal{G}}(\mathsf{free}(\flat\psi)) \times \mathrm{St}_\mathcal{G}, \mathrm{St}_\mathcal{G}, \mathrm{Q}_{\flat\psi}, \delta_{\flat\psi},$

$q_{\flat\psi}, \aleph_{\flat\psi}\rangle$ *such that, for all states* $s \in \mathrm{St}_{\mathcal{G}}$ *and assignments* $\chi \in \mathrm{Asg}_{\mathcal{G}}(\mathrm{free}(\flat\psi), s)$, *it holds that* $\mathcal{G}, \chi, s \models \flat\psi$ *iff* $\mathcal{T} \in \mathrm{L}(\mathcal{U}_{\flat\psi}^{\mathcal{G}})$, *where* $\mathcal{T}$ *is the assignment-state encoding for* χ.

PROOF. A first step in the construction of the UCT $\mathcal{U}_{\flat\psi}^{\mathcal{G}}$ is to consider the UCW $\mathcal{U}_{\psi} \triangleq \langle 2^{\mathrm{AP}}, \mathrm{Q}_{\psi}, \delta_{\psi}, \mathrm{Q}_{0\psi}, \aleph_{\psi}\rangle$ obtained by dualizing the NBW resulting from the application of the classic Vardi-Wolper construction to the LTL formula $\neg\psi$ [Vardi and Wolper 1986]. Observe that $\mathrm{L}(\mathcal{U}_{\psi}) = \mathrm{L}(\psi)$, i.e., $\mathcal{U}_{\psi}$ recognizes all infinite words on the alphabet 2^{AP} that satisfy the LTL formula ψ. Then, define the components of $\mathcal{U}_{\flat\psi}^{\mathcal{G}} \triangleq \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathrm{free}(\flat\psi)) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\flat\psi}, \delta_{\flat\psi}, q_{0\flat\psi}, \aleph_{\flat\psi}\rangle$ as follows:

— $\mathrm{Q}_{\flat\psi} \triangleq \{q_{0\flat\psi}\} \cup \mathrm{Q}_{\psi}$, with $q_{0\flat\psi} \notin \mathrm{Q}_{\psi}$;
— $\delta_{\flat\psi}(q_{0\flat\psi}, (\mathsf{v}, s)) \triangleq \bigwedge_{q \in \mathrm{Q}_{0\psi}} \delta_{\flat\psi}(q, (\mathsf{v}, s))$, for all $(\mathsf{v}, s) \in \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathrm{free}(\flat\psi)) \times \mathrm{St}_{\mathcal{G}}$;
— $\delta_{\flat\psi}(q, (\mathsf{v}, s)) \triangleq \bigwedge_{q' \in \delta_{\psi}(q, \lambda_{\mathcal{G}}(s))}(\tau_{\mathcal{G}}(s, \mathsf{v} \circ \zeta_{\flat}), q')$, for all $q \in \mathrm{Q}_{\psi}$ and $(\mathsf{v}, s) \in \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathrm{free}(\flat\psi)) \times \mathrm{St}_{\mathcal{G}}$;
— $\aleph_{\flat\psi} \triangleq \aleph_{\psi}$.

Intuitively, the UCT $\mathcal{U}_{\flat\psi}^{\mathcal{G}}$ simply runs the UCW $\mathcal{U}_{\psi}$ on the branch of the encoding individuated by the assignment in input. Thus, it is easy to see that, for all states $s \in \mathrm{St}_{\mathcal{G}}$ and assignments $\chi \in \mathrm{Asg}_{\mathcal{G}}(\mathrm{free}(\flat\psi), s)$, it holds that $\mathcal{G}, \chi, s \models \flat\psi$ iff $\mathcal{T} \in \mathrm{L}(\mathcal{U}_{\flat\psi}^{\mathcal{G}})$, where $\mathcal{T}$ is the assignment-state encoding for χ. □

Now, to describe our modified technique, we introduce a new concept of encoding regarding the elementary dependence maps over strategies.

Definition 5.11 (*Elementary Dependence-State Encoding*). Let $\mathcal{G}$ be a CGS, $s \in \mathrm{St}_{\mathcal{G}}$ one of its states, and $\theta \in \mathrm{EDM}_{\mathrm{Str}_{\mathcal{G}}(s)}(\wp)$ an elementary dependence map over strategies for a quantification prefix $\wp \in \mathrm{Qnt}(\mathrm{V})$ over the set $\mathrm{V} \subseteq \mathrm{Var}$. Then, a $(\mathrm{DM}_{\mathrm{Ac}_{\mathcal{G}}}(\wp) \times \mathrm{St}_{\mathcal{G}})$-labeled $\mathrm{St}_{\mathcal{G}}$-tree $\mathcal{T} \triangleq \langle \mathrm{T}, \mathsf{u} \rangle$, where $\mathrm{T} \triangleq \{\rho_{\geq 1} : \rho \in \mathrm{Trk}_{\mathcal{G}}(s)\}$, is an *elementary dependence-state encoding* for θ if it holds that $\mathsf{u}(t) \triangleq (\widetilde{\theta}(s \cdot t), \mathsf{lst}(s \cdot t))$, for all $t \in \mathrm{T}$.

Observe that there exists a unique elementary dependence-state encoding for each elementary dependence map over strategies.

In the next lemma, we show how to handle locally the strategy quantifications on each state of the model, by simply using a quantification over actions, which is modeled by the choice of an action dependence map. Intuitively, we guess in the labeling what is the right part of the dependence map over strategies for each node of the tree and then verify that, for all assignments of universal variables, the corresponding complete assignment satisfies the inner formula.

LEMMA 5.12 (SL[1G] SENTENCE AUTOMATON). *Let* $\mathcal{G}$ *be a* CGS *and* $\wp\flat\psi$ *an* SL[1G] *principal sentence without principal subsentences. Then, there exists a* UCT $\mathcal{U}_{\wp\flat\psi}^{\mathcal{G}} \triangleq \langle \mathrm{DM}_{\mathrm{Ac}_{\mathcal{G}}}(\wp) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\wp\flat\psi}, \delta_{\wp\flat\psi}, q_{0\wp\flat\psi}, \aleph_{\wp\flat\psi}\rangle$ *such that, for all states* $s \in \mathrm{St}_{\mathcal{G}}$ *and elementary dependence maps over strategies* $\theta \in \mathrm{EDM}_{\mathrm{Str}_{\mathcal{G}}(s)}(\wp)$, *it holds that* $\mathcal{G}, \theta(\chi), s \models_{\mathrm{E}} \flat\psi$, *for all* $\chi \in \mathrm{Asg}_{\mathcal{G}}([[\wp]], s)$, *iff* $\mathcal{T} \in \mathrm{L}(\mathcal{U}_{\wp\flat\psi}^{\mathcal{G}})$, *where* $\mathcal{T}$ *is the elementary dependence-state encoding for* θ.

PROOF. By Lemma 5.10 of SL[1G] goal automaton, there is an UCT $\mathcal{U}_{\flat\psi}^{\mathcal{G}} = \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathrm{free}(\flat\psi)) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\flat\psi}, \delta_{\flat\psi}, q_{0\flat\psi}, \aleph_{\flat\psi}\rangle$ such that, for all states $s \in \mathrm{St}_{\mathcal{G}}$ and assignments $\chi \in \mathrm{Asg}_{\mathcal{G}}(\mathrm{free}(\flat\psi), s)$, it holds that $\mathcal{G}, \chi, s \models \flat\psi$ iff $\mathcal{T} \in \mathrm{L}(\mathcal{U}_{\flat\psi}^{\mathcal{G}})$, where $\mathcal{T}$ is the assignment-state encoding for χ.

Now, transform $\mathcal{U}_{\flat\psi}^{\mathcal{G}}$ into the new UCT $\mathcal{U}_{\wp\flat\psi}^{\mathcal{G}} \triangleq \langle \mathrm{DM}_{\mathrm{Ac}_{\mathcal{G}}}(\wp) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\wp\flat\psi}, \delta_{\wp\flat\psi}, q_{0\wp\flat\psi}, \aleph_{\wp\flat\psi}\rangle$, with $\mathrm{Q}_{\wp\flat\psi} \triangleq \mathrm{Q}_{\flat\psi}$, $q_{0\wp\flat\psi} \triangleq q_{0\flat\psi}$, and $\aleph_{\wp\flat\psi} \triangleq \aleph_{\flat\psi}$, which is used to handle the quantification prefix $\wp$ atomically, where the transition function is defined as follows: $\delta_{\wp\flat\psi}(q, (\theta, s)) \triangleq$

$\bigwedge_{v \in \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}([\![\wp]\!])} \delta_{\flat\psi}(q, (\theta(v), s))$, for all $q \in \mathrm{Q}_{\wp\flat\psi}$ and $(\theta, s) \in \mathrm{DM}_{\mathrm{Ac}_{\mathcal{G}}}(\wp) \times \mathrm{St}_{\mathcal{G}}$. Intuitively, $\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi}$ reads an action dependence map θ on each node of the input tree $\mathcal{T}$ labeled with a state s of $\mathcal{G}$ and simulates the execution of the transition function $\delta_{\flat\psi}(q, (v, s))$ of $\mathcal{U}^{\mathcal{G}}_{\flat\psi}$, for each possible valuation $v = \theta(v')$ on $\mathrm{free}(\flat\psi)$ obtained from θ by a universal valuation $v' \in \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}([\![\wp]\!])$. It is important to observe that we cannot move the component set $\mathrm{DM}_{\mathrm{Ac}_{\mathcal{G}}}(\wp)$ from the input alphabet to the states of $\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi}$, by making a related guessing of the dependence map θ in the transition function, since we have to ensure that all states in a given node of the tree $\mathcal{T}$, i.e., in each track of the original model $\mathcal{G}$, make the same choice for θ.

Finally, it remains to prove that, for all states $s \in \mathrm{St}_{\mathcal{G}}$ and elementary dependence map over strategies $\theta \in \mathrm{EDM}_{\mathrm{Str}_{\mathcal{G}}(s)}(\wp)$, it holds that $\mathcal{G}, \theta(\chi), s \models_{\mathrm{E}} \flat\psi$, for all $\chi \in \mathrm{Asg}_{\mathcal{G}}([\![\wp]\!], s)$, iff $\mathcal{T} \in \mathrm{L}(\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi})$, where $\mathcal{T}$ is the elementary dependence-state encoding for θ.

[Only if]. Suppose that $\mathcal{G}, \theta(\chi), s \models_{\mathrm{E}} \flat\psi$, for all $\chi \in \mathrm{Asg}_{\mathcal{G}}([\![\wp]\!], s)$. Since ψ does not contain principal subsentences, we have that $\mathcal{G}, \theta(\chi), s \models \flat\psi$. So, due to the property of $\mathcal{U}^{\mathcal{G}}_{\flat\psi}$, it follows that there exists an assignment-state encoding $\mathcal{T}_{\chi} \in \mathrm{L}(\mathcal{U}^{\mathcal{G}}_{\flat\psi})$, which implies the existence of an $(\mathrm{St}_{\mathcal{G}} \times \mathrm{Q}_{\flat\psi})$-tree R_{χ} that is an accepting run for $\mathcal{U}^{\mathcal{G}}_{\flat\psi}$ on $\mathcal{T}_{\chi}$. At this point, let $\mathrm{R} \triangleq \bigcup_{\chi \in \mathrm{Asg}_{\mathcal{G}}([\![\wp]\!], s)} \mathrm{R}_{\chi}$ be the union of all runs. Then, due to the particular definition of the transition function of $\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi}$, it is not hard to see that R is an accepting run for $\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi}$ on $\mathcal{T}$. Hence, $\mathcal{T} \in \mathrm{L}(\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi})$.

[If]. Suppose that $\mathcal{T} \in \mathrm{L}(\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi})$. Then, there exists an $(\mathrm{St}_{\mathcal{G}} \times \mathrm{Q}_{\wp\flat\psi})$-tree R that is an accepting run for $\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi}$ on $\mathcal{T}$. Now, for each $\chi \in \mathrm{Asg}_{\mathcal{G}}([\![\wp]\!], s)$, let R_{χ} be the run for $\mathcal{U}^{\mathcal{G}}_{\flat\psi}$ on the assignment-state encoding $\mathcal{T}_{\chi}$ for $\theta(\chi)$. Due to the particular definition of the transition function of $\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi}$, it is easy to see that $\mathrm{R}_{\chi} \subseteq \mathrm{R}$. Thus, since R is accepting, we have that R_{χ} is accepting as well. So, $\mathcal{T}_{\chi} \in \mathrm{L}(\mathcal{U}^{\mathcal{G}}_{\flat\psi})$. At this point, due to the property of $\mathcal{U}^{\mathcal{G}}_{\flat\psi}$, it follows that $\mathcal{G}, \theta(\chi), s \models \flat\psi$. Now, since ψ does not contain principal subsentences, we have that $\mathcal{G}, \theta(\chi), s \models_{\mathrm{E}} \flat\psi$, for all $\chi \in \mathrm{Asg}_{\mathcal{G}}([\![\wp]\!], s)$. $\square$

At this point, we can prove the following theorem that is at the base of the elementary model-checking procedure for SL[1G].

THEOREM 5.13 (SL[1G] SENTENCE AUTOMATON). *Let $\mathcal{G}$ be a CGS, $s \in \mathrm{St}_{\mathcal{G}}$ one of its states, and $\wp\flat\psi$ an SL[1G] principal sentence without principal subsentences. Then, there exists an NPT $\mathcal{N}^{\mathcal{G},s}_{\wp\flat\psi}$ such that $\mathcal{G}, \varnothing, s \models \wp\flat\psi$ iff $\mathrm{L}(\mathcal{N}^{\mathcal{G},s}_{\wp\flat\psi}) \neq \emptyset$.*

PROOF. As in the general case of SL sentence automaton, we have to ensure that the state labeling of nodes of the elementary dependence-state encoding is coherent with the node itself. To do this, we apply Theorem 5.4 of APT direction projection with distinguished direction s to the UPT $\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi}$ derived by Lemma 5.12 of the SL[1G] sentence automaton, thus obtaining the required NPT $\mathcal{N}^{\mathcal{G},s}_{\wp\flat\psi}$.

[Only if]. Suppose that $\mathcal{G}, \varnothing, s \models \wp\flat\psi$. By Corollary 4.25 of SL[1G] elementariness, it means that $\mathcal{G}, \varnothing, s \models_{\mathrm{E}} \wp\flat\psi$. Then, by Definition 4.11 of SL[NG] elementary semantics, there exists an elementary dependence map $\theta \in \mathrm{EDM}_{\mathrm{Str}_{\mathcal{G}}(s)}(\wp)$ such that $\mathcal{G}, \theta(\chi), s \models_{\mathrm{E}} \flat\psi$, for all $\chi \in \mathrm{Asg}_{\mathcal{G}}([\![\wp]\!], s)$. Thus, by Lemma 5.12, we have that $\mathcal{T} \in \mathrm{L}(\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi})$, where $\mathcal{T}$ is the elementary dependence-state encoding for θ. Hence, by Theorem 5.4, it holds that $\mathrm{L}(\mathcal{N}^{\mathcal{G},s}_{\wp\flat\psi}) \neq \emptyset$.

[If]. Suppose that $\mathrm{L}(\mathcal{N}^{\mathcal{G},s}_{\wp\flat\psi}) \neq \emptyset$. Then, by Theorem 5.4, there exists an $(\mathrm{DM}_{\mathrm{Ac}_{\mathcal{G}}}(\wp) \times \mathrm{St}_{\mathcal{G}})$-labeled $\mathrm{St}_{\mathcal{G}}$-tree $\mathcal{T}$ such that $\mathcal{T} \in \mathrm{L}(\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi})$. Now, it is immediate to see that there is an elementary dependence map $\theta \in \mathrm{EDM}_{\mathrm{Str}_{\mathcal{G}}(s)}(\wp)$ for which $\mathcal{T}$ is the elementary dependence-state encoding. Thus, by Lemma 5.12, we have that $\mathcal{G}, \theta(\chi), s \models_{\mathrm{E}} \flat\psi$, for all $\chi \in \mathrm{Asg}_{\mathcal{G}}([\![\wp]\!], s)$. By Definition 4.11

of SL[NG] elementary semantics, it holds that $\mathcal{G}, \varnothing, s \models_{\mathrm{E}} \wp\flat\psi$. Hence, by Corollary 4.25 of SL[1G] elementariness, it means that $\mathcal{G}, \varnothing, s \models \wp\flat\psi$. □

Finally, we show in the next fundamental theorem the precise complexity of the model-checking for SL[1G].

THEOREM 5.14 (SL[1G] MODEL CHECKING). *The model-checking problem for* SL[1G] *is* PTIME-COMPLETE *w.r.t. the size of the model and* 2EXPTIME-COMPLETE *w.r.t. the size of the specification.*

PROOF. By Theorem 5.13 of SL[1G] sentence automaton, to verify that $\mathcal{G}, \varnothing, s \models \wp\flat\psi$, we simply calculate the emptiness of the NPT $\mathcal{N}^{\mathcal{G},s}_{\wp\flat\psi}$. This automaton is obtained by the operation of direction projection on the UCT $\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi}$, which is in turn derived by the UCT $\mathcal{U}^{\mathcal{G}}_{\flat\psi}$. Now, it is easy to see that the number of states of $\mathcal{U}^{\mathcal{G}}_{\flat\psi}$, and consequently of $\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi}$, is $2^{\mathrm{O}(\mathsf{lng}(\psi))}$. So, $\mathcal{N}^{\mathcal{G},s}_{\wp\flat\psi}$ has $|\mathrm{St}_{\mathcal{G}}| \cdot 2^{2^{\mathrm{O}(\mathsf{lng}(\psi))}}$ states and index $2^{\mathrm{O}(\mathsf{lng}(\psi))}$.

The emptiness problem for such a kind of automaton with n states and index h is solvable in time $\mathrm{O}(n^h)$ [Kupferman and Vardi 1998]. Thus, we get that the time complexity of checking whether $\mathcal{G}, \varnothing, s \models \wp\flat\psi$ is $|\mathrm{St}_{\mathcal{G}}|^{2^{\mathrm{O}(\mathsf{lng}(\psi))}}$. At this point, since we have to do this verification for each possible state $s \in \mathrm{St}_{\mathcal{G}}$ and principal subsentence $\wp\flat\psi \in \mathsf{psnt}(\varphi)$ of the whole SL[1G] specification φ, we derive that the whole bottom-up model-checking procedure requires time $|\mathrm{St}_{\mathcal{G}}|^{2^{\mathrm{O}(\mathsf{lng}(\varphi))}}$. Hence, the membership of the model-checking problem for SL[1G] in PTIME w.r.t. the size of the model and 2EXPTIME w.r.t. the size of the specification directly follows. Finally the thesis is proved, by getting the relative lower bounds from the same problem for ATL* [Alur et al. 2002]. □

6. CONCLUSION

In this paper, we introduced and studied SL as a very powerful logic formalism to reasoning about strategic behaviors of multi-agent concurrent games. In particular, we proved that it subsumes the classical temporal and game logics not using explicit fix-points. As one of the main results about SL, we shown that the relative model-checking problem is decidable but non-elementary hard. As further and interesting practical results, we investigated several of its syntactic fragments. The most appealing one is SL[1G], which is obtained by restricting SL to deal with one temporal goal at a time. Interestingly, SL[1G] strictly extends ATL*, while maintaining all its positive properties. In fact, the model-checking problem is 2EXPTIME-COMPLETE, hence not harder than the one for ATL*. Moreover, although for the sake of space it is not reported in this paper, we shown that it is invariant under bisimulation and decision-unwinding, and consequently, it has the decision-tree model property. The main reason why SL[1G] has all these positive properties is that it satisfies a special model property, which we name "*elementariness*". Informally, this property asserts that all strategy quantifications in a sentence can be reduced to a set of quantifications over actions, which turn out to be easier to handle. We remark that among all SL fragments we investigated, SL[1G] is the only one that satisfies this property. As far as we know, SL[1G] is the first significant proper extension of ATL* having an elementary model-checking problem, and even more, with the same computational complexity. All these positive aspects make us strongly believe that SL[1G] is a valid alternative to ATL* to be used in the field of formal verification for multi-agent concurrent systems.

As another interesting fragment we investigated in this paper, we recall SL[BG]. This logic allows us to express important game-theoretic properties, such as Nash equilibrium, which cannot be defined in SL[1G]. Unfortunately, we do not have an elementary model-checking procedure for it, neither we can exclude it. We leave to investigate this as future work.

Last but not least, from a theoretical point of view, we are convinced that our framework can be used as a unifying basis for logic reasonings about strategic behaviors in multi-agent scenarios and their relationships. In particular, it can be used to study variations and extensions of SL[1G] in a way similar as it has been done in the literature for ATL*. For example, it could be interest-

ing to investigate memoryful SL[1G], by inheriting and extending the "memoryful" concept used for ATL* and CHP-SL and investigated in [Mogavero et al. 2010b] and [Fisman et al. 2010], respectively. Also, we recall that this concept implicitly allows to deal with backwards temporal modalities. As another example, it would be interesting to investigate the graded extension of SL[1G], in a way similar as it has been done in [Bianco et al. 2009; Bianco et al. 2010; Bianco et al. 2012] and [Kupferman et al. 2002; Bonatti et al. 2008] for CTL and μCALCULUS, respectively. We recall that graded quantifiers in branching-time temporal logics allow to count how many equivalent classes of paths satisfy a given property. This concept in SL[1G] would further allow the counting of strategies and so to succinctly check the existence of more than one nonequivalent winning strategy for a given agent, in one shot. We hope to lift to graded SL[1G] questions left open about graded branching-time temporal logic, such as the precise satisfiability complexity of graded full computation tree logic [Bianco et al. 2012].

A. MATHEMATICAL NOTATION

In this short reference appendix, we report the classical mathematical notation and some common definitions that are used along the whole work.

Classic objects. We consider $\mathbb{N}$ as the set of *natural numbers* and $[m,n] \triangleq \{k \in \mathbb{N} : m \leq k \leq n\}$, $[m,n[\triangleq \{k \in \mathbb{N} : m \leq k < n\}$, $]m,n] \triangleq \{k \in \mathbb{N} : m < k \leq n\}$, and $]m,n[\triangleq \{k \in \mathbb{N} : m < k < n\}$ as its *interval* subsets, with $m \in \mathbb{N}$ and $n \in \widehat{\mathbb{N}} \triangleq \mathbb{N} \cup \{\omega\}$, where ω is the *numerable infinity*, i.e., the *least infinite ordinal*. Given a *set* X of *objects*, we denote by $|X| \in \widehat{\mathbb{N}} \cup \{\infty\}$ the *cardinality* of X, i.e., the number of its elements, where ∞ represents a *more than countable* cardinality, and by $2^X \triangleq \{Y : Y \subseteq X\}$ the *powerset* of X, i.e., the set of all its subsets.

Relations. By $R \subseteq X \times Y$ we denote a *relation* between the *domain* $\mathsf{dom}(R) \triangleq X$ and *codomain* $\mathsf{cod}(R) \triangleq Y$, whose *range* is indicated by $\mathsf{rng}(R) \triangleq \{y \in Y : \exists x \in X. (x,y) \in R\}$. We use $R^{-1} \triangleq \{(y,x) \in Y \times X : (x,y) \in R\}$ to represent the *inverse* of R itself. Moreover, by $S \circ R$, with $R \subseteq X \times Y$ and $S \subseteq Y \times Z$, we denote the *composition* of R with S, i.e., the relation $S \circ R \triangleq \{(x,z) \in X \times Z : \exists y \in Y. (x,y) \in R \wedge (y,z) \in S\}$. We also use $R^n \triangleq R^{n-1} \circ R$, with $n \in [1,\omega[$, to indicate the *n-iteration* of $R \subseteq X \times Y$, where $Y \subseteq X$ and $R^0 \triangleq \{(y,y) : y \in Y\}$ is the *identity* on Y. With $R^+ \triangleq \bigcup_{n=1}^{<\omega} R^n$ and $R^* \triangleq R^+ \cup R^0$ we denote, respectively, the *transitive* and *reflexive-transitive closure* of R. Finally, for an *equivalence* relation $R \subseteq X \times X$ on X, we represent with $(X/R) \triangleq \{[x]_R : x \in X\}$, where $[x]_R \triangleq \{x' \in X : (x,x') \in R\}$, the *quotient* set of X w.r.t. R, i.e., the set of all related equivalence *classes* $[\cdot]_R$.

Functions. We use the symbol $Y^X \subseteq 2^{X \times Y}$ to denote the set of *total functions* f from X to Y, i.e., the relations $f \subseteq X \times Y$ such that for all $x \in \mathsf{dom}(f)$ there is exactly one element $y \in \mathsf{cod}(f)$ such that $(x,y) \in f$. Often, we write $f : X \to Y$ and $f : X \rightharpoonup Y$ to indicate, respectively, $f \in Y^X$ and $f \in \bigcup_{X' \subseteq X} Y^{X'}$. Regarding the latter, note that we consider f as a *partial function* from X to Y, where $\mathsf{dom}(f) \subseteq X$ contains all and only the elements for which f is defined. Given a set Z, by $f_{\restriction Z} \triangleq f \cap (Z \times Y)$ we denote the *restriction* of f to the set $X \cap Z$, i.e., the function $f_{\restriction Z} : X \cap Z \rightharpoonup Y$ such that, for all $x \in \mathsf{dom}(f) \cap Z$, it holds that $f_{\restriction Z}(x) = f(x)$. Moreover, with $\varnothing$ we indicate a generic *empty function*, i.e., a function with empty domain. Note that $X \cap Z = \emptyset$ implies $f_{\restriction Z} = \varnothing$. Finally, for two partial functions $f, g : X \rightharpoonup Y$, we use $f \uplus g$ and $f \Cap g$ to represent, respectively, the *union* and *intersection* of these functions defined as follows: $\mathsf{dom}(f \uplus g) \triangleq \mathsf{dom}(f) \cup \mathsf{dom}(g) \setminus \{x \in \mathsf{dom}(f) \cap \mathsf{dom}(g) : f(x) \neq g(x)\}$, $\mathsf{dom}(f \Cap g) \triangleq \{x \in \mathsf{dom}(f) \cap \mathsf{dom}(g) : f(x) = g(x)\}$, $(f \uplus g)(x) = f(x)$ for $x \in \mathsf{dom}(f \uplus g) \cap \mathsf{dom}(f)$, $(f \uplus g)(x) = g(x)$ for $x \in \mathsf{dom}(f \uplus g) \cap \mathsf{dom}(g)$, and $(f \Cap g)(x) = f(x)$ for $x \in \mathsf{dom}(f \Cap g)$.

Words. By X^n, with $n \in \mathbb{N}$, we denote the set of all *n-tuples* of elements from X, by $X^* \triangleq \bigcup_{n=0}^{<\omega} X^n$ the set of *finite words* on the *alphabet* X, by $X^+ \triangleq X^* \setminus \{\varepsilon\}$ the set of *non-empty words*,

and by X^ω the set of *infinite words*, where, as usual, $\varepsilon \in \mathrm{X}^*$ is the *empty word*. The *length* of a word $w \in \mathrm{X}^\infty \triangleq \mathrm{X}^* \cup \mathrm{X}^\omega$ is represented with $|w| \in \widehat{\mathbb{N}}$. By $(w)_i$ we indicate the *i-th letter* of the finite word $w \in \mathrm{X}^+$, with $i \in [0, |w|[$. Furthermore, by $\mathsf{fst}(w) \triangleq (w)_0$ (resp., $\mathsf{lst}(w) \triangleq (w)_{|w|-1}$), we denote the *first* (resp., *last*) letter of w. In addition, by $(w)_{\leq i}$ (resp., $(w)_{>i}$), we indicate the *prefix* up to (resp., *suffix* after) the letter of index i of w, i.e., the finite word built by the first $i+1$ (resp., last $|w| - i - 1$) letters $(w)_0, \ldots, (w)_i$ (resp., $(w)_{i+1}, \ldots, (w)_{|w|-1}$). We also set, $(w)_{<0} \triangleq \varepsilon$, $(w)_{<i} \triangleq (w)_{\leq i-1}$, $(w)_{\geq 0} \triangleq w$, and $(w)_{\geq i} \triangleq (w)_{>i-1}$, for $i \in [1, |w|[$. Mutatis mutandis, the notations of i-th letter, first, prefix, and suffix apply to infinite words too. Finally, by $\mathsf{pfx}(w_1, w_2) \in \mathrm{X}^\infty$ we denote the *maximal common prefix* of two different words $w_1, w_2 \in \mathrm{X}^\infty$, i.e., the finite word $w \in \mathrm{X}^*$ for which there are two words $w_1', w_2' \in \mathrm{X}^\infty$ such that $w_1 = w \cdot w_1'$, $w_2 = w \cdot w_2'$, and $\mathsf{fst}(w_1') \neq \mathsf{fst}(w_2')$. By convention, we set $\mathsf{pfx}(w, w) \triangleq w$.

Trees. For a set Δ of objects, called *directions*, a Δ-*tree* is a set $\mathrm{T} \subseteq \Delta^*$ closed under prefix, i.e., if $t \cdot d \in \mathrm{T}$, with $d \in \Delta$, then also $t \in \mathrm{T}$. We say that it is *complete* if it holds that $t \cdot d' \in \mathrm{T}$ whenever $t \cdot d \in \mathrm{T}$, for all $d' < d$, where $< \subseteq \Delta \times \Delta$ is an a priori fixed strict total order on the set of directions that is clear from the context. Moreover, it is *full* if $\mathrm{T} = \Delta^*$. The elements of T are called *nodes* and the empty word ε is the *root* of T. For every $t \in \mathrm{T}$ and $d \in \Delta$, the node $t \cdot d \in \mathrm{T}$ is a *successor* of t in T. The tree is b-*bounded* if the maximal number b of its successor nodes is finite, i.e., $b = \max_{t \in \mathrm{T}} |\{t \cdot d \in \mathrm{T} : d \in \Delta\}| < \omega$. A *branch* of the tree is an infinite word $w \in \Delta^\omega$ such that $(w)_{\leq i} \in \mathrm{T}$, for all $i \in \mathbb{N}$. For a finite set Σ of objects, called *symbols*, a Σ-*labeled* Δ-*tree* is a quadruple $\langle \Sigma, \Delta, \mathrm{T}, \mathsf{v} \rangle$, where T is a Δ-tree and $\mathsf{v} : \mathrm{T} \to \Sigma$ is a *labeling function*. When Δ and Σ are clear from the context, we call $\langle \mathrm{T}, \mathsf{v} \rangle$ simply a (labeled) tree.

B. PROOFS OF SECTION ??

In this appendix, we report the proofs of lemmas needed to prove the elementariness of SL[1G]. Before this, we describe two relevant properties that link together dependence maps of a given quantification prefix with those of the dual one. These properties report, in the dependence maps framework, what is known to hold, in an equivalent way, for first and second order logic. In particular, they result to be two key points towards a complete understanding of the strategy quantifications of our logic.

The first of these properties enlighten the fact that two arbitrary dual dependence maps θ and $\overline{\theta}$ always share a common valuation v. To better understand this concept, consider for instance the functions θ_1 and $\overline{\theta}_6$ of the examples illustrated just after Definition 4.5 of dependence maps. Then, it is easy to see that the valuation $\mathsf{v} \in \mathrm{Val}_\mathrm{D}(\mathrm{V})$ with $\mathsf{v}(\mathsf{x}) = \mathsf{v}(\mathsf{y}) = 1$ and $\mathsf{v}(\mathsf{z}) = 0$ resides in both the ranges of θ_1 and $\overline{\theta}_6$, i.e., $\mathsf{v} \in \mathsf{rng}(\theta_1) \cap \mathsf{rng}(\overline{\theta}_6)$.

LEMMA B.1 (DEPENDENCE INCIDENCE). *Let $\wp \in \mathrm{Qnt}(\mathrm{V})$ be a quantification prefix over a set of variables $\mathrm{V} \subseteq \mathrm{Var}$ and D a generic set. Moreover, let $\theta \in \mathrm{DM}_\mathrm{D}(\wp)$ and $\overline{\theta} \in \mathrm{DM}_\mathrm{D}(\overline{\wp})$ be two dependence maps. Then, there exists a valuation $\mathsf{v} \in \mathrm{Val}_\mathrm{D}(\mathrm{V})$ such that $\mathsf{v} = \theta(\mathsf{v}_{\restriction \llbracket \wp \rrbracket}) = \overline{\theta}(\mathsf{v}_{\restriction \llbracket \overline{\wp} \rrbracket})$.*

PROOF. W.l.o.g., suppose that $\wp$ starts with an existential quantifier. If this is not the case, the dual prefix $\overline{\wp}$ necessarily satisfies the above requirement, so, we can simply shift our reasoning on it.

The whole proof proceeds by induction on the alternation number $\mathsf{alt}(\wp)$ of $\wp$. As base case, if $\mathsf{alt}(\wp) = 0$, we define $\mathsf{v} \triangleq \theta(\varnothing)$, since $\llbracket \wp \rrbracket = \emptyset$. Obviously, it holds that $\mathsf{v} = \theta(\mathsf{v}_{\restriction \llbracket \wp \rrbracket}) = \overline{\theta}(\mathsf{v}_{\restriction \llbracket \overline{\wp} \rrbracket})$, due to the fact that $\mathsf{v}_{\restriction \llbracket \wp \rrbracket} = \varnothing$ and $\mathsf{v}_{\restriction \llbracket \overline{\wp} \rrbracket} = \mathsf{v}$. Now, as inductive case, suppose that the statement is true for all prefixes $\wp' \in \mathrm{Qnt}(\mathrm{V}')$ with $\mathsf{alt}(\wp') = n$, where $\mathrm{V}' \subset \mathrm{V}$. Then, we prove that it is true for all prefixes $\wp \in \mathrm{Qnt}(\mathrm{V})$ with $\mathsf{alt}(\wp) = n + 1$ too. To do this, we have to uniquely split $\wp = \wp' \cdot \wp''$ into the two prefixes $\wp' \in \mathrm{Qnt}(\mathrm{V}')$ and $\wp'' \in \mathrm{Qnt}(\mathrm{V} \setminus \mathrm{V}')$ such that $\mathsf{alt}(\wp') = n$ and $\mathsf{alt}(\wp'') = 0$. At this point, the following two cases can arise.

—If n is even, it is immediate to see that $\langle\langle\wp''\rangle\rangle = \emptyset$. So, consider the dependence maps $\theta' \in \mathrm{DM_D}(\wp')$ and $\overline{\theta'} \in \mathrm{DM_D}(\overline{\wp'})$ such that $\theta'(v_{\upharpoonright[\![\wp']\!]}) = \theta(v)_{\upharpoonright V'}$ and $\overline{\theta'}(\overline{v}) = \overline{\theta}(\overline{v})_{\upharpoonright V'}$, for all valuations $v \in \mathrm{Val_D}([\![\wp]\!])$ and $\overline{v} \in \mathrm{Val_D}([\![\overline{\wp}]\!]) = \mathrm{Val_D}([\![\overline{\wp'}]\!])$. By the inductive hypothesis, there exists a valuation $v' \in \mathrm{Val_D}(V')$ such that $v' = \theta'(v'_{\upharpoonright[\![\wp']\!]}) = \overline{\theta'}(v'_{\upharpoonright[\![\overline{\wp'}]\!]})$. So, set $v \triangleq \overline{\theta}(v'_{\upharpoonright[\![\overline{\wp}]\!]})$.

—If n is odd, it is immediate to see that $[\![\wp'']\!] = \emptyset$. So, consider the dependence maps $\theta' \in \mathrm{DM_D}(\wp')$ and $\overline{\theta'} \in \mathrm{DM_D}(\overline{\wp'})$ such that $\theta'(v) = \theta(v)_{\upharpoonright V'}$ and $\overline{\theta'}(\overline{v}_{\upharpoonright[\![\overline{\wp'}]\!]}) = \overline{\theta}(\overline{v})_{\upharpoonright V'}$, for all valuations $v \in \mathrm{Val_D}([\![\wp]\!]) = \mathrm{Val_D}([\![\wp']\!])$ and $\overline{v} \in \mathrm{Val_D}([\![\overline{\wp}]\!])$. By the inductive hypothesis, there exists a valuation $v' \in \mathrm{Val_D}(V')$ such that $v' = \theta'(v'_{\upharpoonright[\![\wp']\!]}) = \overline{\theta'}(v'_{\upharpoonright[\![\overline{\wp'}]\!]})$. So, set $v \triangleq \theta(v'_{\upharpoonright[\![\wp]\!]})$.

Now, it is easy to see that in both cases the valuation v satisfies the thesis, i.e., $v = \theta(v_{\upharpoonright[\![\wp]\!]}) = \overline{\theta}(v_{\upharpoonright[\![\overline{\wp}]\!]})$. $\square$

The second property we are going to prove describes the fact that, if all dependence maps θ of a given prefix $\wp$, for a dependent specific universal valuation v, share a given property then there is a dual dependence maps $\overline{\theta}$ that has the same property, for all universal valuations $\overline{v}$. To have a better understanding of this idea, consider again the examples reported just after Definition 4.5 and let $P \triangleq \{(0,0,1),(0,1,0)\} \subset \mathrm{Val_D}(V)$, where the triple (l,m,n) stands for the valuation that assigns l to x, m to y, and n to z. Then, it is easy to see that all ranges of the dependence maps θ_i for $\wp$ intersect P, i.e., for all $i \in [0,3]$, there is $v \in \mathrm{Val_D}([\![\wp]\!])$ such that $\theta_i(v) \in P$. Moreover, consider the dual dependence maps $\overline{\theta}_2$ for $\overline{\wp}$. Then, it is not hard to see that $\overline{\theta}_2(\overline{v}) \in P$, for all $\overline{v} \in \mathrm{Val_D}([\![\overline{\wp}]\!])$.

LEMMA B.2 (DEPENDENCE DUALIZATION). *Let $\wp \in \mathrm{Qnt}(V)$ be a quantification prefix over a set of variables $V \subseteq \mathrm{Var}$, D a generic set, and $P \subseteq \mathrm{Val_D}(V)$ a set of valuations of V over D. Moreover, suppose that, for all dependence maps $\theta \in \mathrm{DM_D}(\wp)$, there is a valuation $v \in \mathrm{Val_D}([\![\wp]\!])$ such that $\theta(v) \in P$. Then, there exists a dependence map $\overline{\theta} \in \mathrm{DM_D}(\overline{\wp})$ such that, for all valuations $\overline{v} \in \mathrm{Val_D}([\![\overline{\wp}]\!])$, it holds that $\overline{\theta}(\overline{v}) \in P$.*

PROOF. The proof easily proceeds by induction on the length of the prefix $\wp$. As base case, when $|\wp| = 0$, we have that $\mathrm{DM_D}(\wp) = \mathrm{DM_D}(\overline{\wp}) = \{\varnothing\}$, i.e., the only possible dependence maps is the empty function, which means that the statement is vacuously verified. As inductive case, we have to distinguish between two cases, as follows.

— $\wp = \langle\langle x \rangle\rangle \cdot \wp'$.

As first thing, note that $[\![\wp]\!] = [\![\wp']\!]$ and, for all elements $e \in D$, consider the projection $P_e \triangleq \{v' \in \mathrm{Val_D}(V(\wp')) : v'[x \mapsto e] \in P\}$ of P on the variable x with value e.

Then, by hypothesis, we can derive that, for all $e \in D$ and $\theta' \in \mathrm{DM_D}(\wp')$, there exists $v' \in \mathrm{Val_D}([\![\wp']\!])$ such that $\theta'(v') \in P_e$. Indeed, let $e \in D$ and $\theta' \in \mathrm{DM_D}(\wp')$, and build the function $\theta : \mathrm{Val_D}([\![\wp]\!]) \to \mathrm{Val_D}(V)$ given by $\theta(v') \triangleq \theta'(v')[x \mapsto e]$, for all $v' \in \mathrm{Val_D}([\![\wp]\!]) = \mathrm{Val_D}([\![\wp']\!])$. It is immediate to see that $\theta \in \mathrm{DM_D}(\wp)$. So, by the hypothesis, there is $v' \in \mathrm{Val_D}([\![\wp]\!])$ such that $\theta(v') \in P$, which implies $\theta'(v')[x \mapsto e] \in P$, and so, $\theta'(v') \in P_e$.

Now, by the inductive hypothesis, for all elements $e \in D$, there exists $\overline{\theta'}_e \in \mathrm{DM_D}(\overline{\wp'})$ such that, for all $\overline{v'} \in \mathrm{Val_D}([\![\overline{\wp'}]\!])$, it holds that $\overline{\theta'}_e(\overline{v'}) \in P_e$, i.e., $\overline{\theta'}_e(\overline{v'})[x \mapsto e] \in P$.

At this point, consider the function $\overline{\theta} : \mathrm{Val_D}([\![\overline{\wp}]\!]) \to \mathrm{Val_D}(V)$ given by $\overline{\theta}(\overline{v}) \triangleq \overline{\theta'}_{\overline{v}(x)}(\overline{v}_{\upharpoonright[\![\overline{\wp'}]\!]})[x \mapsto \overline{v}(x)]$, for all $\overline{v} \in \mathrm{Val_D}([\![\overline{\wp}]\!])$. Then, it is possible to verify that $\overline{\theta} \in \mathrm{DM_D}(\overline{\wp})$. Indeed, for each $y \in [\![\overline{\wp}]\!]$ and $\overline{v} \in \mathrm{Val_D}([\![\overline{\wp}]\!])$, we have that $\overline{\theta}(\overline{v})(y) = \overline{\theta'}_{\overline{v}(x)}(\overline{v}_{\upharpoonright[\![\overline{\wp'}]\!]})[x \mapsto \overline{v}(x)](y)$. Now, if $y = x$ then $\overline{\theta}(\overline{v})(y) = \overline{v}(y)$. Otherwise, since $\overline{\theta'}_{\overline{v}(x)}$ is a dependence map, it holds that $\overline{\theta}(\overline{v})(y) = \overline{\theta'}_{\overline{v}(x)}(\overline{v}_{\upharpoonright[\![\overline{\wp'}]\!]})(y) = \overline{v}_{\upharpoonright[\![\overline{\wp'}]\!]}(y) = \overline{v}(y)$. So, Item 1 of Definition 4.5 of dependence maps is verified. It only remains to prove Item 2. Let $y \in \langle\langle\overline{\wp}\rangle\rangle$ and $\overline{v_1}, \overline{v_2} \in \mathrm{Val_D}([\![\overline{\wp}]\!])$,

with $\overline{v_1}_{\upharpoonright \mathrm{Dep}(\overline{\wp}, y)} = \overline{v_2}_{\upharpoonright \mathrm{Dep}(\overline{\wp}, y)}$. It is immediate to see that $x \in \mathrm{Dep}(\overline{\wp}, y)$, so, $\overline{v_1}(x) = \overline{v_2}(x)$, which implies that $\overline{\theta'}_{\overline{v_1}(x)} = \overline{\theta'}_{\overline{v_2}(x)}$. At this point, again for the fact that $\overline{\theta'}_{\overline{v}(x)}$ is a dependence map, for each $\overline{v} \in \mathrm{Val}_{\mathrm{D}}(\llbracket \overline{\wp} \rrbracket)$, we have that $\overline{\theta'}_{\overline{v_1}(x)}(\overline{v_1}_{\upharpoonright \llbracket \overline{\wp'} \rrbracket})(y) = \overline{\theta'}_{\overline{v_2}(x)}(\overline{v_2}_{\upharpoonright \llbracket \overline{\wp'} \rrbracket})(y)$. Thus, it holds that $\overline{\theta}(\overline{v_1})(y) = \overline{\theta'}_{\overline{v_1}(x)}(\overline{v_1}_{\upharpoonright \llbracket \overline{\wp'} \rrbracket})[x \mapsto \overline{v_1}(x)](y) = \overline{\theta'}_{\overline{v_2}(x)}(\overline{v_2}_{\upharpoonright \llbracket \overline{\wp'} \rrbracket})[x \mapsto \overline{v_2}(x)](y) = \overline{\theta}(\overline{v_2})(y)$.

Finally, it is enough to observe that, by construction, $\overline{\theta}(\overline{v}) \in \mathrm{P}$, for all $\overline{v} \in \mathrm{Val}_{\mathrm{D}}(\llbracket \overline{\wp} \rrbracket)$, since $\overline{\theta'}_{\overline{v}(x)}(\overline{v}_{\upharpoonright \llbracket \overline{\wp'} \rrbracket}) \in \mathrm{P}_{\overline{v}(x)}$. Thus, the thesis holds for this case.

— $\wp = \llbracket x \rrbracket \cdot \wp'$.
We first show that there exists $e \in \mathrm{D}$ such that, for all $\theta' \in \mathrm{DM}_{\mathrm{D}}(\wp')$, there is $v' \in \mathrm{Val}_{\mathrm{D}}(\llbracket \wp' \rrbracket)$ for which $\theta'(v') \in \mathrm{P}_e$ holds, where the set P_e is defined as above.

To do this, suppose by contradiction that, for all $e \in \mathrm{D}$, there is a $\theta'_e \in \mathrm{DM}_{\mathrm{D}}(\wp')$ such that, for all $v' \in \mathrm{Val}_{\mathrm{D}}(\llbracket \wp' \rrbracket)$, it holds that $\theta'_e(v') \notin \mathrm{P}_e$. Also, consider the function $\theta : \mathrm{Val}_{\mathrm{D}}(\llbracket \wp \rrbracket) \to \mathrm{Val}_{\mathrm{D}}(\mathrm{V})$ given by $\theta(v) \triangleq \theta'_{v(x)}(v_{\upharpoonright \llbracket \wp' \rrbracket})[x \mapsto v(x)]$, for all $v \in \mathrm{Val}_{\mathrm{D}}(\llbracket \wp \rrbracket)$. Then, is possible to verify that $\theta \in \mathrm{DM}_{\mathrm{D}}(\wp)$. Indeed, for each $y \in \llbracket \wp \rrbracket$ and $v \in \mathrm{Val}_{\mathrm{D}}(\llbracket \wp \rrbracket)$, we have that $\theta(v)(y) = \theta'_{v(x)}(v_{\upharpoonright \llbracket \wp' \rrbracket})[x \mapsto v(x)](y)$. Now, if $y = x$ then $\theta(v)(y) = v(y)$. Otherwise, since $\theta'_{v(x)}$ is a dependence map, it holds that $\theta(v)(y) = \theta'_{v(x)}(v_{\upharpoonright \llbracket \wp' \rrbracket})(y) = v_{\upharpoonright \llbracket \wp' \rrbracket}(y) = v(y)$. So, Item 1 of Definition 4.5 of dependence maps is verified. It only remains to prove Item 2. Let $y \in \langle\langle \wp \rangle\rangle$ and $v_1, v_2 \in \mathrm{Val}_{\mathrm{D}}(\llbracket \wp \rrbracket)$, with $v_{1 \upharpoonright \mathrm{Dep}(\wp, y)} = v_{2 \upharpoonright \mathrm{Dep}(\wp, y)}$. It is immediate to see that $x \in \mathrm{Dep}(\wp, y)$, so, $v_1(x) = v_2(x)$, which implies that $\theta'_{v_1(x)} = \theta'_{v_2(x)}$. At this point, again for the fact that $\theta'_{v(x)}$ is a dependence map, for each $v \in \mathrm{Val}_{\mathrm{D}}(\llbracket \wp \rrbracket)$, we have that $\theta'_{v_1(x)}(v_{1 \upharpoonright \llbracket \wp' \rrbracket})(y) = \theta'_{v_2(x)}(v_{2 \upharpoonright \llbracket \wp' \rrbracket})(y)$. Thus, it holds that $\theta(v_1)(y) = \theta'_{v_1(x)}(v_{1 \upharpoonright \llbracket \wp' \rrbracket})[x \mapsto v_1(x)](y) = \theta'_{v_2(x)}(v_{2 \upharpoonright \llbracket \wp' \rrbracket})[x \mapsto v_2(x)](y) = \theta(v_2)(y)$. Now, by the contradiction hypothesis, we have that $\theta(v) \notin \mathrm{P}$, for all $v \in \mathrm{Val}(\llbracket \wp \rrbracket)$, since $\theta'_{v(x)}(v_{\upharpoonright \llbracket \wp' \rrbracket}) \notin \mathrm{P}_{v(x)}$, which is in evident contradiction with the hypothesis.

At this point, by the inductive hypothesis, there exists $\overline{\theta'} \in \mathrm{DM}_{\mathrm{D}}(\overline{\wp'})$ such that, for all $\overline{v'} \in \mathrm{Val}_{\mathrm{D}}(\llbracket \overline{\wp'} \rrbracket)$, it holds that $\overline{\theta'}(\overline{v'}) \in \mathrm{P}_e$, i.e., $\overline{\theta'}(\overline{v'})[x \mapsto e] \in \mathrm{P}$.

Finally, build the function $\overline{\theta} : \mathrm{Val}_{\mathrm{D}}(\llbracket \overline{\wp} \rrbracket) \to \mathrm{Val}_{\mathrm{D}}(\mathrm{V})$ given by $\overline{\theta}(\overline{v}) \triangleq \overline{\theta'}(\overline{v})[x \mapsto e]$, for all $\overline{v} \in \mathrm{Val}_{\mathrm{D}}(\llbracket \overline{\wp} \rrbracket) = \mathrm{Val}_{\mathrm{D}}(\llbracket \overline{\wp'} \rrbracket)$. It is immediate to see that $\overline{\theta} \in \mathrm{DM}_{\mathrm{D}}(\overline{\wp})$. Moreover, for all valuations $\overline{v} \in \mathrm{Val}_{\mathrm{D}}(\llbracket \overline{\wp} \rrbracket)$, it holds that $\overline{\theta}(\overline{v}) \in \mathrm{P}$. Thus, the thesis holds for this case too.

Hence, we have done with the proof of the lemma. $\quad \square$

At this point, we are able to give the proofs of Lemma 4.9 of adjoint dependence maps, Lemma 4.21 of dependence-vs-valuation duality, and Lemma 4.23 of encasement characterization.

LEMMA B.3 (ADJOINT DEPENDENCE MAPS). *Let $\wp \in \mathrm{Qnt}(\mathrm{V})$ be a quantification prefix over a set of variables $\mathrm{V} \subseteq \mathrm{Var}$, D and T two generic sets, and $\theta : \mathrm{Val}_{\mathrm{T} \to \mathrm{D}}(\llbracket \wp \rrbracket) \to \mathrm{Val}_{\mathrm{T} \to \mathrm{D}}(\mathrm{V})$ and $\widetilde{\theta} : \mathrm{T} \to (\mathrm{Val}_{\mathrm{D}}(\llbracket \wp \rrbracket) \to \mathrm{Val}_{\mathrm{D}}(\mathrm{V}))$ two functions such that $\widetilde{\theta}$ is the adjoint of θ. Then, $\theta \in \mathrm{DM}_{\mathrm{T} \to \mathrm{D}}(\wp)$ iff, for all $t \in \mathrm{T}$, it holds that $\widetilde{\theta}(t) \in \mathrm{DM}_{\mathrm{D}}(\wp)$.*

PROOF. To prove the statement, it is enough to show, separately, that Items 1 and 2 of Definition 4.5 of dependence maps hold for θ if the $\widetilde{\theta}(t)$ satisfies the same items, for all $t \in \mathrm{T}$, and vice versa.

[Item 1, if]. Assume that $\widetilde{\theta}(t)$ satisfies Item 1, for each $t \in \mathrm{T}$, i.e., $\widetilde{\theta}(t)(v)_{\upharpoonright \llbracket \wp \rrbracket} = v$, for all $v \in \mathrm{Val}_{\mathrm{D}}(\llbracket \wp \rrbracket)$. Then, we have that $\widetilde{\theta}(t)(\widehat{g}(t)) = \widehat{g}(t)$, so, $\widetilde{\theta}(t)(\widehat{g}(t))(x) = \widehat{g}(t)(x)$, for all $g \in \mathrm{Val}_{\mathrm{T} \to \mathrm{D}}(\llbracket \wp \rrbracket)$ and $x \in \llbracket \wp \rrbracket$. By hypothesis, we have that $\theta(g)(x)(t) = \widetilde{\theta}(t)(\widehat{g}(t))(x)$, thus $\theta(g)(x)(t) = \widehat{g}(t)(x) = g(x)(t)$, which means that $\theta(g)_{\upharpoonright \llbracket \wp \rrbracket} = g$, for all $g \in \mathrm{Val}_{\mathrm{T} \to \mathrm{D}}(\llbracket \wp \rrbracket)$.

[Item 1, only if]. Assume now that θ satisfies Item 1, i.e., $\theta(g)_{\upharpoonright \llbracket \wp \rrbracket} = g$, for all $g \in \mathrm{Val}_{\mathrm{T} \to \mathrm{D}}(\llbracket \wp \rrbracket)$. Then, we have that $\theta(g)(x)(t) = g(x)(t)$, for all $x \in \llbracket \wp \rrbracket$ and $t \in \mathrm{T}$. By hypothesis, we have

that $\widetilde{\theta}(t)(\widehat{g}(t))(x) = \theta(g)(x)(t)$, so, $\widetilde{\theta}(t)(\widehat{g}(t))(x) = g(x)(t) = \widehat{g}(t)(x)$, which means that $\widetilde{\theta}(t)(\widehat{g}(t))_{\upharpoonright[\![\wp]\!]} = \widehat{g}(t)$. Now, since for each $v \in \mathrm{Val}_D([\![\wp]\!])$, there is an $g \in \mathrm{Val}_{T \to D}([\![\wp]\!])$ such that $\widehat{g}(t) = v$, we obtain that $\widetilde{\theta}(t)(v)_{\upharpoonright[\![\wp]\!]} = v$, for all $v \in \mathrm{Val}_D([\![\wp]\!])$ and $t \in T$.

[Item 2, if]. Assume that $\widetilde{\theta}(t)$ satisfies Item 2, for each $t \in T$, i.e., $\widetilde{\theta}(t)(v_1)(x) = \widetilde{\theta}(t)(v_2)(x)$, for all $v_1, v_2 \in \mathrm{Val}_D([\![\wp]\!])$ and $x \in \langle\!\langle\wp\rangle\!\rangle$ such that $v_1{\upharpoonright}_{\mathrm{Dep}(\wp,x)} = v_2{\upharpoonright}_{\mathrm{Dep}(\wp,x)}$. Then, we have that $\widetilde{\theta}(t)(\widehat{g_1}(t))(x) = \widetilde{\theta}(t)(\widehat{g_2}(t))(x)$, for all $g_1, g_2 \in \mathrm{Val}_{T \to D}([\![\wp]\!])$ such that $g_1{\upharpoonright}_{\mathrm{Dep}(\wp,x)} = g_2{\upharpoonright}_{\mathrm{Dep}(\wp,x)}$. By hypothesis, we have that $\theta(g_1)(x)(t) = \widetilde{\theta}(t)(\widehat{g_1}(t))(x)$ and $\widetilde{\theta}(t)(\widehat{g_2}(t))(x) = \theta(g_2)(x)(t)$, thus $\theta(g_1)(x)(t) = \theta(g_2)(x)(t)$. Hence, $\theta(g_1)(x) = \theta(g_2)(x)$, for all $g_1, g_2 \in \mathrm{Val}_{T \to D}([\![\wp]\!])$ and $x \in \langle\!\langle\wp\rangle\!\rangle$ such that $g_1{\upharpoonright}_{\mathrm{Dep}(\wp,x)} = g_2{\upharpoonright}_{\mathrm{Dep}(\wp,x)}$.

[Item 2, only if]. Assume that θ satisfies Item 2, i.e., $\theta(g_1)(x) = \theta(g_2)(x)$, for all $g_1, g_2 \in \mathrm{Val}_{T \to D}([\![\wp]\!])$ and $x \in \langle\!\langle\wp\rangle\!\rangle$ such that $g_1{\upharpoonright}_{\mathrm{Dep}(\wp,x)} = g_2{\upharpoonright}_{\mathrm{Dep}(\wp,x)}$. Then, we have that $\theta(g_1)(x)(t) = \theta(g_2)(x)(t)$, for all $t \in T$. By hypothesis, we have that $\widetilde{\theta}(t)(\widehat{g_1}(t))(x) = \theta(g_1)(x)(t)$ and $\theta(g_2)(x)(t) = \widetilde{\theta}(t)(\widehat{g_2}(t))(x)$, hence $\widetilde{\theta}(t)(\widehat{g_1}(t))(x) = \widetilde{\theta}(t)(\widehat{g_2}(t))(x)$. Now, since for each $v_1, v_2 \in \mathrm{Val}_D([\![\wp]\!])$, with $v_1{\upharpoonright}_{\mathrm{Dep}(\wp,x)} = v_2{\upharpoonright}_{\mathrm{Dep}(\wp,x)}$, there are $g_1, g_2 \in \mathrm{Val}_{T \to D}([\![\wp]\!])$ such that $\widehat{g_1}(t) = v_1$ and $\widehat{g_2}(t) = v_2$, with $g_1{\upharpoonright}_{\mathrm{Dep}(\wp,x)} = g_2{\upharpoonright}_{\mathrm{Dep}(\wp,x)}$, we obtain that $\widetilde{\theta}(t)(v_1)(x) = \widetilde{\theta}(t)(v_2)(x)$, for all $v_1, v_2 \in \mathrm{Val}_D([\![\wp]\!])$ and $x \in \langle\!\langle\wp\rangle\!\rangle$ such that $v_1{\upharpoonright}_{\mathrm{Dep}(\wp,x)} = v_2{\upharpoonright}_{\mathrm{Dep}(\wp,x)}$. $\square$

LEMMA B.4 (DEPENDENCE-VS-VALUATION DUALITY). *Let $\mathcal{G}$ be a CGS, $s \in \mathrm{St}$ one of its states, $\mathrm{P} \subseteq \mathrm{Pth}(s)$ a set of paths, $\wp \in \mathrm{Qnt}(V)$ a quantification prefix over a set of variables $V \subseteq \mathrm{Var}$, and $\flat \in \mathrm{Bnd}(V)$ a binding. Then, player even wins the TPG $\mathcal{H}(\mathcal{G}, s, \mathrm{P}, \wp, \flat)$ iff player odd wins the dual TPG $\mathcal{H}(\mathcal{G}, s, \mathrm{Pth}(s) \setminus \mathrm{P}, \overline{\wp}, \flat)$.*

PROOF. Let $\mathcal{A}$ and $\overline{\mathcal{A}}$ be, respectively, the two TPAs $\mathcal{A}(\mathcal{G}, s, \wp, \flat)$ and $\mathcal{A}(\mathcal{G}, s, \overline{\wp}, \flat)$. It is easy to observe that $\mathrm{Pos}_{e\mathcal{A}} = \mathrm{Pos}_{e\overline{\mathcal{A}}} = \mathrm{Trk}(s)$. Moreover, it holds that $\mathrm{Pos}_{o\mathcal{A}} = \{\rho \cdot (\mathrm{lst}(\rho), \theta) : \rho \in \mathrm{Trk}(s) \wedge \theta \in \mathrm{DM}_{\mathrm{Ac}}(\wp)\}$ and $\mathrm{Pos}_{o\overline{\mathcal{A}}} = \{\rho \cdot (\mathrm{lst}(\rho), \overline{\theta}) : \rho \in \mathrm{Trk}(s) \wedge \overline{\theta} \in \mathrm{DM}_{\mathrm{Ac}}(\overline{\wp})\}$. We now prove, separately, the two directions of the statement.

[Only if]. Suppose that player even wins the TPG $\mathcal{H}(\mathcal{G}, s, \mathrm{P}, \wp, \flat)$. Then, there exists an even scheme $s_e \in \mathrm{Sch}_{e\mathcal{A}}$ such that, for all odd schemes $s_o \in \mathrm{Sch}_{o\mathcal{A}}$, it holds that $\mathrm{mtc}_{\mathcal{A}}(s_e, s_o) \in \mathrm{P}$. Now, to prove that odd wins the dual TPG $\mathcal{H}(\mathcal{G}, s, \mathrm{Pth}(s) \setminus \mathrm{P}, \overline{\wp}, \flat)$, we have to show that there exists an odd scheme $\overline{s_o} \in \mathrm{Sch}_{o\overline{\mathcal{A}}}$ such that, for all even schemes $\overline{s_e} \in \mathrm{Sch}_{e\overline{\mathcal{A}}}$, it holds that $\mathrm{mtc}_{\overline{\mathcal{A}}}(\overline{s_e}, \overline{s_o}) \in \mathrm{P}$.

To do this, let us first consider a function $z : \mathrm{DM}_{\mathrm{Ac}}(\wp) \times \mathrm{DM}_{\mathrm{Ac}}(\overline{\wp}) \to \mathrm{Val}_{\mathrm{Ac}}(V)$ such that $z(\theta, \overline{\theta}) = \theta(z(\theta, \overline{\theta})_{\upharpoonright[\![\wp]\!]}) = \overline{\theta}(z(\theta, \overline{\theta})_{\upharpoonright[\![\overline{\wp}]\!]})$, for all $\theta \in \mathrm{DM}_{\mathrm{Ac}}(\wp)$ and $\overline{\theta} \in \mathrm{DM}_{\mathrm{Ac}}(\overline{\wp})$. The existence of such a function is ensured by Lemma B.1 on the dependence incidence.

Now, define the odd scheme $\overline{s_o} \in \mathrm{Sch}_{o\overline{\mathcal{A}}}$ in $\overline{\mathcal{A}}$ as follows: $\overline{s_o}(\rho \cdot (\mathrm{lst}(\rho), \overline{\theta})) \triangleq \tau(\mathrm{lst}(\rho), z(\theta, \overline{\theta}) \circ \zeta_\flat)$, for all $\rho \in \mathrm{Trk}(s)$ and $\overline{\theta} \in \mathrm{DM}_{\mathrm{Ac}}(\overline{\wp})$, where $\theta \in \mathrm{DM}_{\mathrm{Ac}}(\wp)$ is such that $s_e(\rho) = (\mathrm{lst}(\rho), \theta)$. Moreover, let $\overline{s_e} \in \mathrm{Sch}_{e\overline{\mathcal{A}}}$ be a generic even scheme in $\overline{\mathcal{A}}$ and consider the derived odd scheme $s_o \in \mathrm{Sch}_{o\mathcal{A}}$ in $\mathcal{A}$ defined as follows: $s_o(\rho \cdot (\mathrm{lst}(\rho), \theta)) \triangleq \tau(\mathrm{lst}(\rho), z(\theta, \overline{\theta}) \circ \zeta_\flat)$, for all $\rho \in \mathrm{Trk}(s)$ and $\theta \in \mathrm{DM}_{\mathrm{Ac}}(\wp)$, where $\overline{\theta} \in \mathrm{DM}_{\mathrm{Ac}}(\overline{\wp})$ is such that $\overline{s_e}(\rho) = (\mathrm{lst}(\rho), \overline{\theta})$.

At this point, it remains only to prove that $\varpi = \overline{\varpi}$, where $\varpi \triangleq \mathrm{mtc}_{\mathcal{A}}(s_e, s_o)$ and $\overline{\varpi} \triangleq \mathrm{mtc}_{\overline{\mathcal{A}}}(\overline{s_e}, \overline{s_o})$. To do this, we proceed by induction on the prefixes of the matches, i.e., we show that $(\varpi)_{\leq i} = (\overline{\varpi})_{\leq i}$, for all $i \in \mathbb{N}$. The base case is immediate by definition of match, since we have that $(\varpi)_{\leq 0} = s = (\overline{\varpi})_{\leq 0}$. Now, as inductive case, suppose that $(\varpi)_{\leq i} = (\overline{\varpi})_{\leq i}$, for $i \in \mathbb{N}$. By the definition of match, we have that $(\varpi)_{i+1} = s_o((\varpi)_{\leq i} \cdot s_e((\varpi)_{\leq i}))$ and $(\overline{\varpi})_{i+1} = \overline{s_o}((\overline{\varpi})_{\leq i} \cdot \overline{s_e}((\overline{\varpi})_{\leq i}))$. Moreover, by the inductive hypothesis, it follows that $s_o((\varpi)_{\leq i} \cdot s_e((\varpi)_{\leq i})) = s_o((\overline{\varpi})_{\leq i} \cdot s_e((\overline{\varpi})_{\leq i}))$. At this point, let $\theta \in \mathrm{DM}_{\mathrm{Ac}}(\wp)$ and $\overline{\theta} \in \mathrm{DM}_{\mathrm{Ac}}(\overline{\wp})$ be two quantification dependence maps such that $s_e((\overline{\varpi})_{\leq i}) = ((\overline{\varpi})_i, \theta)$ and $\overline{s_e}((\overline{\varpi})_{\leq i}) = ((\overline{\varpi})_i, \overline{\theta})$. Consequently, by substituting the values of the even schemes s_e and $\overline{s_e}$, it holds that $(\varpi)_{i+1} = s_o((\overline{\varpi})_{\leq i} \cdot ((\overline{\varpi})_i, \theta))$ and

$(\overline{\varpi})_{i+1} = \overline{s_o}((\overline{\varpi})_{\leq i} \cdot ((\overline{\varpi})_i, \overline{\theta}))$. Furthermore, by the definition of the odd schemes s_o and $\overline{s_o}$, it follows that $s_o((\overline{\varpi})_{\leq i} \cdot ((\overline{\varpi})_i, \theta)) = \tau((\overline{\varpi})_i, z(\theta, \overline{\theta}) \circ \zeta_\flat) = \overline{s_o}((\overline{\varpi})_{\leq i} \cdot ((\overline{\varpi})_i, \overline{\theta}))$. Thus, we have that $(\varpi)_{i+1} = (\overline{\varpi})_{i+1}$, which implies $(\varpi)_{\leq i+1} = (\overline{\varpi})_{\leq i+1}$.

[If]. Suppose that player odd wins the dual TPG $\mathcal{H}(\mathcal{G}, s, \mathrm{Pth}(s) \setminus \mathrm{P}, \overline{\wp}, \flat)$. Then, there exists an odd scheme $\overline{s_o} \in \mathrm{Sch}_{o\overline{\mathcal{A}}}$ such that, for all even schemes $\overline{s_e} \in \mathrm{Sch}_{e\overline{\mathcal{A}}}$, it holds that $\mathrm{mtc}_{\overline{\mathcal{A}}}(\overline{s_e}, \overline{s_o}) \in \mathrm{P}$. Now, to prove that even wins the TPG $\mathcal{H}(\mathcal{G}, s, \mathrm{P}, \wp, \flat)$, we have to show that there exists an even scheme $s_e \in \mathrm{Sch}_{e\mathcal{A}}$ such that, for all odd schemes $s_o \in \mathrm{Sch}_{o\mathcal{A}}$, it holds that $\mathrm{mtc}_{\mathcal{A}}(s_e, s_o) \in \mathrm{P}$.

To do this, let us first consider the two functions $g : \mathrm{Trk}(s) \to 2^{\mathrm{Val}_{\mathrm{Ac}}(V)}$ and $h : \mathrm{Trk}(s) \to 2^{\mathrm{St}}$ such that $g(\rho) \triangleq \{\overline{\theta}(\overline{v}) : \overline{\theta} \in \mathrm{DM}_{\mathrm{Ac}}(\overline{\wp}) \wedge \overline{v} \in \mathrm{Val}_{\mathrm{Ac}}(\llbracket \overline{\wp} \rrbracket) \wedge \overline{s_o}(\rho \cdot (\mathrm{lst}(\rho), \overline{\theta})) = \tau(\mathrm{lst}(\rho), \overline{\theta}(\overline{v}) \circ \zeta_\flat)\}$ and $h(\rho) \triangleq \{\overline{s_o}(\rho \cdot (\mathrm{lst}(\rho), \overline{\theta})) : \overline{\theta} \in \mathrm{DM}_{\mathrm{Ac}}(\overline{\wp})\}$, for all $\rho \in \mathrm{Trk}(s)$. Now, it is easy to see that, for each $\rho \in \mathrm{Trk}(s)$ and $\overline{\theta} \in \mathrm{DM}_{\mathrm{Ac}}(\overline{\wp})$, there is $\overline{v} \in \mathrm{Val}_{\mathrm{Ac}}(\llbracket \overline{\wp} \rrbracket)$ such that $\overline{\theta}(\overline{v}) \in g(\rho)$. Consequently, by Lemma B.2 on dependence dualization, for all $\rho \in \mathrm{Trk}(s)$, there is $\theta_\rho \in \mathrm{DM}_{\mathrm{Ac}}(\wp)$ such that, for each $v \in \mathrm{Val}_{\mathrm{Ac}}(\llbracket \wp \rrbracket)$, it holds that $\theta_\rho(v) \in g(\rho)$, and so, $\tau(\mathrm{lst}(\rho), \theta_\rho(v) \circ \zeta_\flat) \in h(\rho)$.

Now, define the even scheme $s_e \in \mathrm{Sch}_{e\mathcal{A}}$ in $\mathcal{A}$ as follows: $s_e(\rho) \triangleq (\mathrm{lst}(\rho), \theta_\rho)$, for all $\rho \in \mathrm{Trk}(s)$. Moreover, let $s_o \in \mathrm{Sch}_{e\mathcal{A}}$ be a generic odd scheme in $\mathcal{A}$ and consider the derived even scheme $\overline{s_e} \in \mathrm{Sch}_{e\overline{\mathcal{A}}}$ in $\overline{\mathcal{A}}$ defined as follows: $\overline{s_e}(\rho) \triangleq (\mathrm{lst}(\rho), \overline{\theta}_\rho)$, for all $\rho \in \mathrm{Trk}(s)$, where $\overline{\theta}_\rho \in \mathrm{DM}_{\mathrm{Ac}}(\overline{\wp})$ is such that $s_o(\rho \cdot (\mathrm{lst}(\rho), \theta_\rho)) = \overline{s_o}(\rho \cdot (\mathrm{lst}(\rho), \overline{\theta}_\rho))$. The existence of such a dependence map is ensure by the previous membership of the successor of $\mathrm{lst}(\rho)$ in $h(\rho)$.

At this point, it remains only to prove that $\varpi = \overline{\varpi}$, where $\varpi \triangleq \mathrm{mtc}_{\mathcal{A}}(s_e, s_o)$ and $\overline{\varpi} \triangleq \mathrm{mtc}_{\overline{\mathcal{A}}}(\overline{s_e}, \overline{s_o})$. To do this, we proceed by induction on the prefixes of the matches, i.e., we show that $(\varpi)_{\leq i} = (\overline{\varpi})_{\leq i}$, for all $i \in \mathbb{N}$. The base case is immediate by definition of match, since we have that $(\varpi)_{\leq 0} = s = (\overline{\varpi})_{\leq 0}$. Now, as inductive case, suppose that $(\varpi)_{\leq i} = (\overline{\varpi})_{\leq i}$, for $i \in \mathbb{N}$. By the definition of match, we have that $(\varpi)_{i+1} = s_o((\varpi)_{\leq i} \cdot s_e((\varpi)_{\leq i}))$ and $(\overline{\varpi})_{i+1} = \overline{s_o}((\overline{\varpi})_{\leq i} \cdot \overline{s_e}((\overline{\varpi})_{\leq i}))$. Moreover, by the inductive hypothesis, it follows that $s_o((\varpi)_{\leq i} \cdot s_e((\varpi)_{\leq i})) = s_o((\overline{\varpi})_{\leq i} \cdot s_e((\overline{\varpi})_{\leq i}))$. Now, by substituting the values of the even schemes s_e and $\overline{s_e}$, we have that $(\varpi)_{i+1} = s_o((\overline{\varpi})_{\leq i} \cdot ((\overline{\varpi})_i, \theta_{(\overline{\varpi})_{\leq i}}))$ and $(\overline{\varpi})_{i+1} = \overline{s_o}((\overline{\varpi})_{\leq i} \cdot ((\overline{\varpi})_i, \overline{\theta}_{\overline{\varpi}_{\leq i}}))$. At this point, due to the choice of the dependence map $\overline{\theta}_{(\overline{\varpi})_{\leq i}}$, it holds that $s_o((\overline{\varpi})_{\leq i} \cdot ((\overline{\varpi})_i, \theta_{(\overline{\varpi})_{\leq i}})) = \overline{s_o}((\overline{\varpi})_{\leq i} \cdot ((\overline{\varpi})_i, \overline{\theta}_{(\overline{\varpi})_{\leq i}}))$. Thus, we have that $(\varpi)_{i+1} = (\overline{\varpi})_{i+1}$, which implies $(\varpi)_{\leq i+1} = (\overline{\varpi})_{\leq i+1}$. □

LEMMA B.5 (ENCASEMENT CHARACTERIZATION). *Let $\mathcal{G}$ be a CGS, $s \in \mathrm{St}$ one of its states, $\mathrm{P} \subseteq \mathrm{Pth}(s)$ a set of paths, $\wp \in \mathrm{Qnt}(V)$ a quantification prefix over a set of variables $V \subseteq \mathrm{Var}$, and $\flat \in \mathrm{Bnd}(V)$ a binding. Then, the following hold:*

(i) *player even wins $\mathcal{H}(\mathcal{G}, s, \mathrm{P}, \wp, \flat)$ iff P is an encasement w.r.t. $\wp$ and $\flat$;*
(ii) *if player odd wins $\mathcal{H}(\mathcal{G}, s, \mathrm{P}, \wp, \flat)$ then P is not an encasement w.r.t. $\wp$ and $\flat$;*
(iii) *if P is a Borelian set and it is not an encasement w.r.t. $\wp$ and $\flat$ then player odd wins $\mathcal{H}(\mathcal{G}, s, \mathrm{P}, \wp, \flat)$.*

PROOF. *[Item i, only if]*. Suppose that player even wins the TPG $\mathcal{H}(\mathcal{G}, s, \mathrm{P}, \wp, \flat)$. Then, there exists an even scheme $s_e \in \mathrm{Sch}_e$ such that, for all odd schemes $s_o \in \mathrm{Sch}_o$, it holds that $\mathrm{mtc}(s_e, s_o) \in \mathrm{P}$. Now, to prove the statement, we have to show that there exists an elementary dependence map $\theta \in \mathrm{EDM}_{\mathrm{Str}(s)}(\wp)$ such that, for all assignments $\chi \in \mathrm{Asg}(\llbracket \wp \rrbracket, s)$, it holds that $\mathrm{play}(\theta(\chi) \circ \zeta_\flat, s) \in \mathrm{P}$.

To do this, consider the function $w : \mathrm{Trk}(s) \to \mathrm{DM}_{\mathrm{Ac}}(\wp)$ constituting the projection of s_e on the second component of its codomain, i.e., for all $\rho \in \mathrm{Trk}(s)$, it holds that $s_e(\rho) = (\mathrm{lst}(\rho), w(\rho))$. By Lemma 4.9 on adjoint dependence maps, there exists an elementary dependence map $\theta \in \mathrm{EDM}_{\mathrm{Str}(s)}(\wp)$ for which w is the adjoint, i.e., $w = \widetilde{\theta}$. Moreover, let $\chi \in \mathrm{Asg}(\llbracket \wp \rrbracket, s)$ be a generic assignment and consider the derived odd scheme $s_o \in \mathrm{Sch}_o$ defined ad follows: $s_o(\rho \cdot (\mathrm{lst}(\rho), \theta')) = \tau(\mathrm{lst}(\rho), \theta'(\widehat{\chi}(\rho)) \circ \zeta_\flat)$, for all $\rho \in \mathrm{Trk}(s)$ and $\theta' \in \mathrm{DM}_{\mathrm{Ac}}(\wp)$.

At this point, it remains only to prove that $\pi = \varpi$, where $\pi \triangleq \mathsf{play}(\theta(\chi) \circ \zeta_\flat, s)$ and $\varpi \triangleq \mathsf{mtc}(\mathsf{s_e}, \mathsf{s_o})$. To do this, we proceed by induction on the prefixes of both the play and the match, i.e., we show that $(\pi)_{\leq i} = (\varpi)_{\leq i}$, for all $i \in \mathbb{N}$. The base case is immediate by definition, since we have that $(\pi)_{\leq 0} = s = (\varpi)_{\leq 0}$. Now, as inductive case, suppose that $(\pi)_{\leq i} = (\varpi)_{\leq i}$, for $i \in \mathbb{N}$. On one hand, by the definition of match, we have that $(\varpi)_{i+1} = \mathsf{s_o}((\varpi)_{\leq i} \cdot \mathsf{s_e}((\varpi)_{\leq i}))$, from which, by substituting the value of the even scheme $\mathsf{s_e}$, we derive $(\varpi)_{i+1} = \mathsf{s_o}((\varpi)_{\leq i} \cdot ((\varpi)_i, \widetilde{\theta}((\varpi)_{\leq i})))$. On the other hand, by the definition of play, we have that $(\pi)_{i+1} = \tau((\pi)_i, \widetilde{\theta}((\pi)_{\leq i})(\widehat{\chi}((\pi)_{\leq i})) \circ \zeta_\flat)$, from which, by using the definition of the odd scheme $\mathsf{s_o}$, we derive $(\pi)_{i+1} = \mathsf{s_o}((\pi)_{\leq i} \cdot ((\pi)_i, \widetilde{\theta}((\pi)_{\leq i})))$. Then, by the inductive hypothesis, we have that $(\varpi)_{i+1} = \mathsf{s_o}((\varpi)_{\leq i} \cdot ((\varpi)_i, \widetilde{\theta}((\varpi)_{\leq i}))) = \mathsf{s_o}((\pi)_{\leq i} \cdot ((\pi)_i, \widetilde{\theta}((\pi)_{\leq i}))) = (\pi)_{i+1}$, which implies $(\varpi)_{\leq i+1} = (\pi)_{\leq i+1}$.

[Item i, if]. Suppose that P is an encasement w.r.t. $\wp$ and $\flat$. Then, there exists an elementary dependence map $\theta \in \mathrm{EDM}_{\mathrm{Str}(s)}(\wp)$ such that, for all assignments $\chi \in \mathrm{Asg}(\llbracket \wp \rrbracket, s)$, it holds that $\mathsf{play}(\theta(\chi) \circ \zeta_\flat, s) \in \mathrm{P}$. Now, to prove the statement, we have to show that there exists an even scheme $\mathsf{s_e} \in \mathrm{Sch_e}$ such that, for all odd schemes $\mathsf{s_o} \in \mathrm{Sch_o}$, it holds that $\mathsf{mtc}(\mathsf{s_e}, \mathsf{s_o}) \in \mathrm{P}$.

To do this, consider the even scheme $\mathsf{s_e} \in \mathrm{Sch_e}$ defined as follows: $\mathsf{s_e}(\rho) \triangleq (\mathsf{lst}(\rho), \widetilde{\theta}(\rho))$, for all $\rho \in \mathrm{Trk}(s)$. Observe that, by Lemma 4.9 on adjoint dependence maps, the definition is well-formed. Moreover, let $\mathsf{s_o} \in \mathrm{Sch_o}$ be a generic odd scheme and consider a derived assignment $\chi \in \mathrm{Asg}(\llbracket \wp \rrbracket, s)$ satisfying the following property: $\widehat{\chi}(\rho) \in \{\mathsf{v} \in \mathrm{Val_{Ac}}(\llbracket \wp \rrbracket) : \mathsf{s_o}(\rho \cdot (\mathsf{lst}(\rho), \widetilde{\theta}(\rho))) = \tau(\mathsf{lst}(\rho), \widetilde{\theta}(\mathsf{v}) \circ \zeta_\flat)\}$, for all $\rho \in \mathrm{Trk}(s)$.

At this point, it remains only to prove that $\pi = \varpi$, where $\pi \triangleq \mathsf{play}(\theta(\chi) \circ \zeta_\flat, s)$ and $\varpi \triangleq \mathsf{mtc}(\mathsf{s_e}, \mathsf{s_o})$. To do this, we proceed by induction on the prefixes of both the play and the match, i.e., we show that $(\pi)_{\leq i} = (\varpi)_{\leq i}$, for all $i \in \mathbb{N}$. The base case is immediate by definition, since we have that $(\pi)_{\leq 0} = s = (\varpi)_{\leq 0}$. Now, as inductive case, suppose that $(\pi)_{\leq i} = (\varpi)_{\leq i}$, for $i \in \mathbb{N}$. On one hand, by the definition of match, we have that $(\varpi)_{i+1} = \mathsf{s_o}((\varpi)_{\leq i} \cdot \mathsf{s_e}((\varpi)_{\leq i}))$, from which, by the definition of the even scheme $\mathsf{s_e}$, we derive $(\varpi)_{i+1} = \mathsf{s_o}((\varpi)_{\leq i} \cdot ((\varpi)_i, \widetilde{\theta}((\varpi)_{\leq i})))$. On the other hand, by the definition of play, we have that $(\pi)_{i+1} = \tau((\pi)_i, \widetilde{\theta}((\pi)_{\leq i})(\widehat{\chi}((\pi)_{\leq i})) \circ \zeta_\flat)$, from which, by the choice of the assignment χ, we derive $(\pi)_{i+1} = \mathsf{s_o}((\pi)_{\leq i} \cdot ((\pi)_i, \widetilde{\theta}((\pi)_{\leq i})))$. Then, by the inductive hypothesis, we have that $(\varpi)_{i+1} = \mathsf{s_o}((\varpi)_{\leq i} \cdot ((\varpi)_i, \widetilde{\theta}((\varpi)_{\leq i}))) = \mathsf{s_o}((\pi)_{\leq i} \cdot ((\pi)_i, \widetilde{\theta}((\pi)_{\leq i}))) = (\pi)_{i+1}$, which implies $(\varpi)_{\leq i+1} = (\pi)_{\leq i+1}$.

[Item ii]. If player odd wins the TPG $\mathcal{H}(\mathcal{G}, s, \mathrm{P}, \wp, \flat)$, we have that player even does not win the same game. Consequently, by Item i, it holds that P is not an encasement w.r.t. $\wp$ and $\flat$.

[Item iii]. If P is not an encasement w.r.t. $\wp$ and $\flat$, by Item i, we have that player even does not win the TPG $\mathcal{H}(\mathcal{G}, s, \mathrm{P}, \wp, \flat)$. Now, since P is Borelian, by the determinacy theorem [Martin 1975; Martin 1985], it holds that player odd wins the same game. $\square$

REFERENCES

ALUR, R., HENZINGER, T., AND KUPFERMAN, O. 2002. Alternating-Time Temporal Logic. *Journal of the ACM 49,* 5, 672–713.

BIANCO, A., MOGAVERO, F., AND MURANO, A. 2009. Graded Computation Tree Logic. In *IEEE Symposium on Logic in Computer Science'09.* IEEE Computer Society, 342–351.

BIANCO, A., MOGAVERO, F., AND MURANO, A. 2010. Graded Computation Tree Logic with Binary Coding. In *EACSL Annual Conference on Computer Science Logic'10.* LNCS 6247. Springer, 125–139.

BIANCO, A., MOGAVERO, F., AND MURANO, A. 2012. Graded Computation Tree Logic. *ACM Transactions On Computational Logic 13,* 3. Under publication.

BONATTI, P., LUTZ, C., MURANO, A., AND VARDI, M. 2008. The Complexity of Enriched Mu-Calculi. *Logical Methods in Computer Science 4,* 3, 1–27.

BRIHAYE, T., LOPES, A., LAROUSSINIE, F., AND MARKEY, N. 2009. ATL with Strategy Contexts and Bounded Memory. In *Symposium on Logical Foundations of Computer Science'09*. LNCS 5407. Springer, 92–106.

BÜCHI, J. 1962. On a Decision Method in Restricted Second-Order Arithmetic. In *International Congress on Logic, Methodology, and Philosophy of Science'62*. Stanford University Press, 1–11.

CHATTERJEE, K., HENZINGER, T., AND PITERMAN, N. 2007. Strategy Logic. In *International Conference on Concurrency Theory'07*. LNCS 4703. Springer, 59–73.

CHATTERJEE, K., HENZINGER, T., AND PITERMAN, N. 2010. Strategy Logic. *Information and Computation 208,* 6, 677–693.

CLARKE, E. AND EMERSON, E. 1981. Design and Synthesis of Synchronization Skeletons Using Branching-Time Temporal Logic. In *Logic of Programs'81*. LNCS 131. Springer, 52–71.

CLARKE, E., GRUMBERG, O., AND PELED, D. 2002. *Model Checking.* MIT Press.

COSTA, A. D., LAROUSSINIE, F., AND MARKEY, N. 2010a. ATL with Strategy Contexts: Expressiveness and Model Checking. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science'10*. LIPIcs 8. 120–132.

COSTA, A. D., LAROUSSINIE, F., AND MARKEY, N. 2010b. Personal communication.

EBBINGHAUS, H. AND FLUM, J. 1995. *Finite Model Theory.* Springer-Verlag.

ELGAARD, J., KLARLUND, N., AND MØLLER, A. 1998. MONA 1.x: New Techniques for WS1S and WS2S. In *Computer Aided Verification'98*. LNCS 1427. Springer, 516–520.

EMERSON, E. 1990. Temporal and Modal Logic. In *Handbook of Theoretical Computer Science (vol. B)*. MIT Press, 995–1072.

EMERSON, E. AND HALPERN, J. 1986. "Sometimes" and "Not Never" Revisited: On Branching Versus Linear Time. *Journal of the ACM 33,* 1, 151–178.

FAGIN, R., HALPERN, J., MOSES, Y., AND VARDI, M. 1995. *Reasoning about Knowledge.* MIT Press.

FINKBEINER, B. AND SCHEWE, S. 2010. Coordination Logic. In *EACSL Annual Conference on Computer Science Logic'10*. LNCS 6247. Springer, 305–319.

FISMAN, D., KUPFERMAN, O., AND LUSTIG, Y. 2010. Rational Synthesis. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems'10*. LNCS 6015. Springer, 190–204.

GRÄDEL, E., THOMAS, W., AND WILKE, T. 2002. *Automata, Logics, and Infinite Games: A Guide to Current Research.* LNCS 2500. Springer-Verlag.

HODGES, W. 1993. *Model theory.* Encyclopedia of Mathematics and its Applications. Cambridge University Press.

JAMROGA, W. AND VAN DER HOEK, W. 2004. Agents that Know How to Play. *Fundamenta Informaticae 63,* 2-3, 185–219.

KELLER, R. 1976. Formal verification of parallel programs. *Communication of the ACM 19,* 7, 371–384.

KOZEN, D. 1983. Results on the Propositional mu-Calculus. *Theoretical Computer Science 27,* 3, 333–354.

KRIPKE, S. 1963. Semantical Considerations on Modal Logic. *Acta Philosophica Fennica 16*, 83–94.

KUPFERMAN, O., SATTLER, U., AND VARDI, M. 2002. The Complexity of the Graded μ-Calculus. In *Conference on Automated Deduction'02*. LNCS 2392. Springer, 423–437.

KUPFERMAN, O. AND VARDI, M. 1998. Weak Alternating Automata and Tree Automata Emptiness. In *ACM Symposium on Theory of Computing'98*. 224–233.

KUPFERMAN, O., VARDI, M., AND WOLPER, P. 2000. An Automata Theoretic Approach to Branching-Time Model Checking. *Journal of the ACM 47,* 2, 312–360.

KUPFERMAN, O., VARDI, M., AND WOLPER, P. 2001. Module Checking. *Information and Computation 164,* 2, 322–344.

MARTIN, A. 1975. Borel Determinacy. *Annals of Mathematics 102,* 2, 363–371.

MARTIN, A. 1985. A Purely Inductive Proof of Borel Determinacy. In *Symposia in Pure Mathematics'82*. Recursion Theory. American Mathematical Society and Association for Symbolic Logic, 303–308.

MOGAVERO, F., MURANO, A., PERELLI, G., AND VARDI, M. 2012. A Decidable Fragment of Strategy Logic. Submitted.

MOGAVERO, F., MURANO, A., AND VARDI, M. 2010a. Reasoning About Strategies. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science'10*. LIPIcs 8. 133–144.

MOGAVERO, F., MURANO, A., AND VARDI, M. 2010b. Relentful Strategic Reasoning in Alternating-Time Temporal Logic. In *International Conference on Logic for Programming Artificial Intelligence and Reasoning'10*. LNAI 6355. Springer, 371–387.

MULLER, D., SAOUDI, A., AND SCHUPP, P. 1988. Weak Alternating Automata Give a Simple Explanation of Why Most Temporal and Dynamic Logics are Decidable in Exponential Time. In *IEEE Symposium on Logic in Computer Science'88*. IEEE Computer Society, 422–427.

MULLER, D. AND SCHUPP, P. 1987. Alternating Automata on Infinite Trees. *Theoretical Computer Science 54,* 2-3, 267–276.

MULLER, D. AND SCHUPP, P. 1995. Simulating Alternating Tree Automata by Nondeterministic Automata: New Results and New Proofs of Theorems of Rabin, McNaughton, and Safra. *Theoretical Computer Science 141,* 1-2, 69–107.

PAULY, M. 2002. A Modal Logic for Coalitional Power in Games. *Journal of Logic and Computation 12,* 1, 149–166.

PERRIN, D. AND PIN, J. 2004. *Infinite Words.* Pure and Applied Mathematics Series, vol. 141. Elsevier.

PINCHINAT, S. 2007. A Generic Constructive Solution for Concurrent Games with Expressive Constraints on Strategies. In *International Symposium on Automated Technology for Verification and Analysis'07*. LNCS 4762. Springer, 253–267.

PNUELI, A. 1977. The Temporal Logic of Programs. In *Foundation of Computer Science'77*. 46–57.

QUEILLE, J. AND SIFAKIS, J. 1981. Specification and Verification of Concurrent Programs in Cesar. In *International Symposium on Programming'81*. LNCS 137. Springer, 337–351.

RABIN, M. 1969. Decidability of Second-Order Theories and Automata on Infinite Trees. *Transactions of the American Mathematical Society 141*, 1–35.

SCHEWE, S. 2008. ATL* Satisfiability is 2ExpTime-Complete. In *International Colloquium on Automata, Languages and Programming'08*. LNCS 5126. Springer, 373–385.

SISTLA, A. 1983. Theoretical Issues in the Design and Cerification of Distributed Systems. Ph.D. thesis, Harvard University, Cambridge, MA, USA.

SISTLA, A., VARDI, M., AND WOLPER, P. 1987. The Complementation Problem for Büchi Automata with Applications to Temporal Logic. *Theoretical Computer Science 49*, 217–237.

THOMAS, W. 1990. Automata on Infinite Objects. In *Handbook of Theoretical Computer Science (vol. B)*. MIT Press, 133–191.

VARDI, M. 1988. A Temporal Fixpoint Calculus. In *ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages'88*. 250–259.

VARDI, M. 1996. Why is Modal Logic So Robustly Decidable? In *Descriptive Complexity and Finite Models'96*. American Mathematical Society, 149–184.

VARDI, M. AND WOLPER, P. 1986. An Automata-Theoretic Approach to Automatic Program Verification. In *IEEE Symposium on Logic in Computer Science'86*. IEEE Computer Society, 332–344.