Reasoning About Strategies: On the Model-Checking Problem

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

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 multiagent 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 nonelementary 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 multiagent systems.

In this article, we introduce and study a more general strategy logic, denoted SL, for reasoning about strategies in multiagent concurrent games. As a key aspect, strategies in SL are not intrinsically glued to a specific agent, but an explicit binding operator allows an agent to bind to a strategy variable. This allows agents to share strategies or reuse one previously adopted. We prove that SL strictly 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 NONELEMENTARY. This negative result has spurred us to investigate syntactic fragments of SL, strictly subsuming ATL*, with the hope of obtaining an elementary model-checking problem. Among others, we introduce and 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. Intuitively, for a goal, we mean a sequence of bindings, one for each agent, followed by an LTL formula. 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 NONELEMENTARY-HARD, strengthening the corresponding result for SL. Regarding SL[BG], we show that it includes CHP-SL and its model-checking is decidable with a 2EXPTIMElower-bound.

It is worth enlightening that to achieve the positive results about SL[1G], we introduce a fundamental property of the semantics of this logic, called *behavioral*, which allows to strongly simplify the reasoning about strategies. Indeed, in a nonbehavioral logic such as SL[BG] and the subsuming ones, to satisfy a formula, one has to take into account that a move of an agent, at a given moment of a play, may depend on the moves taken by any agent in another counterfactual play.

Categories and Subject Descriptors: F.3.1 [**Logics and Meanings of Programs**]: Specifying and Verifying and Reasoning about Programs—*Specification techniques*; F.4.1 [**Mathematical Logic and Formal Languages**]: Mathematical Logic—*Modal logic; Temporal logic*

This work is partially based on the paper by Mogavero et al. [2010a],which appeared in FSTTCS'10.
This work is partially supported by the FP7 European Union project 600958-SHERPA and the Embedded System Cup Project, B25B09090100007 (POR Campania FS 2007/2013, asse IV e asse V).
Authors' addresses: F. Mogavero, A. Murano, and G. Perelli, Via Cinthia, Complesso Monte S. Angelo, I-80126, Napoli, Italy; M. Y. Vardi, 6100 Main St. Houston, TX 77005-1892, U.S.A.

DOI: http://dx.doi.org/10.1145/2631917

General Terms: Theory, Specification, Verification

Additional Key Words and Phrases: Strategy logic, model checking, behavioral strategies

ACM Reference Format:
Fabio Mogavero, Aniello Murano, Giuseppe Perelli, and Moshe Y. Vardi. 2014. Reasoning about strategies: On the model-checking problem. ACM Trans. Comput. Logic 15, 4, Article 34 (November 2014), 47 pages.
DOI: http://dx.doi.org/10.1145/2631917

1. INTRODUCTION

In system design, *model checking* is a well-established formal method that allows for global system correctness to be checked automatically [Clarke and Emerson 1981; Queille and Sifakis 1981; Clarke et al. 2002]. In such a framework, to check whether a system satisfies a required property, we describe its structure in a mathematical model (such as Kripke structures [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 formally check 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 continuously 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 (see also Ferrante et al. [2008], Aminof et al. [2013], and Jamroga and Murano [2014] for recent works in this field).

Starting from the study of module checking, researchers have looked for logics focusing on the strategic behavior of agents in multiagent systems [Agotnes et al. 2007; Alur et al. 2002; Pauly 2002; Jamroga and van der Hoek 2004; Walther et al. 2007; Bulling and Jamroga 2014; Benerecetti et al. 2013]. One of the most important developments in this field is alternating-time temporal logic (ATL*), introduced by 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 "$\langle\langle \mathrm{A} \rangle\rangle$" and "$[\![\mathrm{A}]\!]$," 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 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 (CGSS) [Alur et al. 2002], which model interacting processes. Given a CGS $\mathcal{G}$ and a set A of agents, the ATL* formula $\langle\langle \mathrm{A} \rangle\rangle \psi$ is satisfied at a state s of $\mathcal{G}$ if there is a set of strategies for agents in A such that, no matter which 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, whereas 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 $\langle\langle \{\alpha, \beta\} \rangle\rangle \mathsf{G}\, \neg \mathit{fail}$, where G is the classical LTL temporal operator "*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. It turns out that both the model-checking and the satisfiability problems are elementarily decidable and, precisely, 2EXPTIME-COMPLETE [Alur et al. 2002; Schewe 2008].

Despite its powerful expressiveness, ATL* suffers from a strong limitation because of the fact that strategies are only treated implicitly, through modalities that refer

to games between competing coalitions. To overcome this problem, Chatterjee et al. [2007] introduced strategy logic (CHP-SL), 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 $\exists \mathsf{x}.\forall \mathsf{y}.\psi(\mathsf{x},\mathsf{y})$, that is, "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, Chatterjee et al. [2007] show 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 nonelementary algorithm for it, both in the size of system and formula, has been provided, leaving as an open question whether an algorithm with a better complexity exists. The complementary question about the decidability of the satisfiability problem for CHP-SL was also left open and it is not addressed in other papers apart our preliminary work [Mogavero et al. 2010a].

Although 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 multiagent 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 (SL) as a more general framework than CHP-SL, for explicit reasoning about strategies in multiagent 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 three agents (α, β, γ), the previous ATL* formula $\langle\langle\{\alpha, \beta\}\rangle\rangle \mathsf{G}\, \neg \mathit{fail}$ can be translated in the SL formula $\langle\langle \mathsf{x}\rangle\rangle\langle\langle \mathsf{y}\rangle\rangle[\![\mathsf{z}]\!](\alpha, \mathsf{x})(\beta, \mathsf{y})(\gamma, \mathsf{z})(\mathsf{G}\, \neg \mathit{fail})$. 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 $\langle\langle \mathsf{x}\rangle\rangle[\![\mathsf{z}]\!](\alpha, \mathsf{x})(\beta, \mathsf{x})(\gamma, \mathsf{z})(\mathsf{G}\, \neg \mathit{fail})$. Furthermore, we may vary the structure of the game by changing the way the quantifiers alternate, as in the formula $\langle\langle \mathsf{x}\rangle\rangle[\![\mathsf{z}]\!]\langle\langle \mathsf{y}\rangle\rangle(\alpha, \mathsf{x})(\beta, \mathsf{y})(\alpha, \mathsf{z})(\mathsf{G}\, \neg \mathit{fail})$. In this case, x remains uniform with respect to 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 $\langle\langle \mathsf{x}\rangle\rangle\langle\langle \mathsf{y}\rangle\rangle[\![\mathsf{z}]\!]\langle\langle \mathsf{w}\rangle\rangle(\alpha, \mathsf{x})(\beta, \mathsf{y})(\gamma, \mathsf{z})(\mathsf{G}\, (\gamma, \mathsf{w})\mathsf{G}\, \neg \mathit{fail})$. The last examples intuitively show that SL is an 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 article about SL, we show that the model-checking problem is nonelementarily 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]. Because the operations of projection required by the elimination of quantifications on strategies, which induce an exponential blowup at any step, the overall size of the required automaton is nonelementary 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, with respect to the size of the model, and NonElementary, with respect to the size of the specification. Hence, the algorithm we propose is computationally not harder than the best one known for CHP-Sl and, notably, a nonelementary data complexity improvement (i.e., with respect to the size of the model). The latter is remarkable as in formal verification often the specification is very small (or even constant) with respect to the system. Moreover, we prove that our problem has a nonelementary 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 want 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, differently from the first two, we introduced the last in a very recent paper and investigated it with respect to the satisfiability problem [Mogavero et al. 2012]. These fragments 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 NonElementary-hard, and thus we enforce the corresponding result for Sl. Finally, by observing that Sl[ng] includes Sl[bg], we have that the model-checking problem for the latter is decidable with a 2ExpTime lower-bound, given by Sl[1g] model-checking complexity. The problem of determine its precise complexity is left open here.

To achieve all positive results about Sl[1g], we introduce a fundamental property of the semantics of this logic, called *behavioral,*[1] which allows us to strongly simplify the reasoning about strategies by reducing it to a set of reasonings about the agent's choices (formally, the agent's *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

[1]We use this term as it has a direct correspondence with the "behavioral" concept used in game theory [Myerson 1997; Mogavero et al. 2013, 2014a].

strategies, as for the whole SL, but only their values on the histories of interest. In other words, by means of the behavioral property, a one-shot second-order quantification over strategies can be eliminated in favor of a progressive first-order quantification over the agent's choices. Technically, to describe the behavioral property, we make use of the machinery of the Skolem dependence function, which defines a Skolemization procedure for SL, inspired by the one in first-order logic (see Hodges [2001]).

By means of behavioral, 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 behavioral 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 and ATL* to incorporate more powerful strategic constructs. Among them, we recall alternating-time μCalculus (AμCalculus) [Alur et al. 2002], game logic (GL) [Alur et al. 2002], quantified decision modality μCalculus (QDμCalculus) [Pinchinat 2007], coordination logic (CL) [Finkbeiner and Schewe 2010], ATLwith plausibility (ATL$^+$) [Bulling et al. 2008], ATLwith Irrevocable strategies (IATL) [Agotnes et al. 2007], memoryful ATL* (mATL*) [Mogavero et al. 2010b], basic strategy-interaction logic (BSIL) [Wang et al. 2011], temporal Cooperation Logic (TCL) [Huang et al. 2013], alternating-time temporal logic with explicit actions (ATLEA) [Herzig et al. 2013], and some extensions of ATL* considered in Brihaye et al. [2009]. AμCalculus and QDμCalculus 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 QDμCalculus 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, nor does it use the extensions of ATL* introduced in Brihaye et al. [2009], which consider restrictions on the memory for strategy quantifiers. ATL$^+$enables to express rationality assumptions of intelligent agents in ATL. In IATL, the semantics of the logic ATL is changed in a way that, in the evaluation of the goal, agents can be forced to keep the strategy they have chosen in the past to reach the state where a goal is evaluated. mATL* enriches ATL* by giving the ability to agents to "relent" and change their goals and strategies depending on the history of the play. BSIL allows to specify behaviors of a system that can cooperate with several strategies of the environment for different requirements. TCL extends ATL by allowing successive definitions of agent strategies, with the aim of using the collaborative power of groups of agents to enforce different temporal objectives. ATLEA introduces explicit actions in the logic ATL to reason about abilities of agents under commitments to play precise actions. Thus, all of the aforementioned logics are different from SL, which we recall it aims to be a minimal but powerful logic to reason about strategic behavior in multiagent systems.

At roughly the same time we have conceived SL, another generalization of ATL*, named ATL* with strategy contexts, which results to be very expressive but a proper sublogic of SL, has been considered in Da Costa et al. [2010a] (see also Da Costa et al. [2012] and Laroussinie and Markey [2013] for more recent works). In this logic, a quantification over strategies does not reset the strategies previously quantified but

allows to maintain them in a particular context 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 article. We want to remark that our nonelementary hardness proof about the SL model-checking problem is inspired by and improves a proof for the logic described in [Da Costa et al. 2010a] and communicated to us by the authors [Da Costa et al. 2010b].

Recently, several extensions of SL have been also investigated. updating strategy logic (U-SL) has been considered in Chareton et al. [2013] where, in addition to SL, an agent can refine its own strategies by means of an "unbinder" operator, which explicitly deletes the binding of a strategy to an agent. In Belardinelli [2014], an epistemic extension of SL with modal operators for individual knowledge has been considered, showing that the complexity of model checking for this logic is not worse than the one for (nonepistemic) SL. Last but not least, in Čermák et al. [2014], a BDD-based model checker for the verification of systems against specifications expressed in the epistemic extension of SL, named MCMAS-SLK, has been introduced.

Finally, works worthy of mention are those handling the synthesis question of specifications expressed in the logic CHP-SL, as well as logics related to SL. Among the others, we report the works of Chatterjee et al. [2014] and Fisman et al. [2010].

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 the model-checking problem for the full SL is nonelementary.

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. In Section 3, we show the nonelementary lower bound for the model-checking problem. In Section 4, we start the study of few syntactic and semantic SL fragments and introduce the concepts of Skolem dependence function and behavioral satisfiability. In Section 5, we describe the model-checking automata-theoretic procedures for all SL fragments. Finally, in Section 6, we give some concluding observation. Note that in the accompanying electronic appendix, we recall some standard mathematical notation and basic definitions that are used in the article. 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 nontrivial or immediate mathematical concept. The article is self-contained. All missing proofs in the main body of the work are reported in the electronic 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, 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 allowed us to write Boolean combinations and nesting of complex predicates, linked together by some common strategic choices, 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 Section 2.1, we recall the definition of concurrent game structure, used to interpret SL, along with some useful example. In Section 2.2, we introduce the syntax. Then, in Section 2.3, we give, among the others, the notions of strategy, assignment, and play, which are finally used, in Section 2.4, to define the semantics of the logic.

2.1. Underlying Framework

As a semantic framework for our logic language, we use a graph-based model for multiplayer 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 multiagent 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) 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 nonempty sets of atomic propositions and agents, Ac and St are enumerable nonempty 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, that is, functions from Ag to Ac representing the choices of an action for each agent.[2] 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 because 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 the literature [Pinchinat 2007], this simplifying choice does not result in a limitation of our semantics framework and allows 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 no longer make their definition explicit. In addition, we use the italic letters p, a, c, and s, possibly with indexes, as metavariables on, respectively, the atomic propositions $\mathsf{p}, \mathsf{q}, \ldots$ in AP, the agents $\alpha, \beta, \gamma, \ldots$ in Ag, the actions $0, 1, \ldots$ in Ac, and the states $\mathsf{s}, \ldots$ 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.

[2] 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.

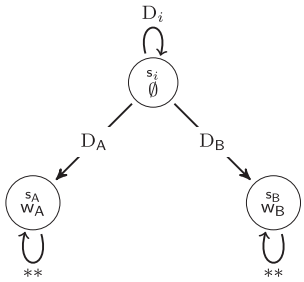


Fig. 1. The CGS $\mathcal{G}_{PRS}$.

Now, to get attitude to the introduced semantic framework, let us describe some running examples of simple concurrent games. 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) 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 $**$ 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 accordingly 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 between A and B. 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. The labeling is given by $\lambda(\mathsf{s}_i) \triangleq \emptyset$, $\lambda(\mathsf{s}_\mathsf{A}) \triangleq \{\mathsf{w}_\mathsf{A}\}$, and $\lambda(\mathsf{s}_\mathsf{B}) \triangleq \{\mathsf{w}_\mathsf{B}\}$. Moreover, the transition function is defined 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.

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

Example 2.3 (*Prisoner's Dilemma*). In the prisoner's dilemma (PD), 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 cooperates (i.e., remains silent), the defector goes free and the silent accomplice 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 they both defect, they both go to jail. Note that no one of the two prisoners knows about the choice made by the other. This tricky situation can be modeled by the CGS $\mathcal{G}_{PD} \triangleq \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}, \mathrm{St}, \lambda, \tau, \mathsf{s}_i \rangle$ depicted in Figure 2, where the agents "Accomplice-1" and "Accomplice-2" in $\mathrm{Ag} \triangleq \{\mathsf{A}_1, \mathsf{A}_2\}$ can chose an action in $\mathrm{Ac} \triangleq \{\mathsf{C}, \mathsf{D}\}$, which stand for "cooperation" and "defection," respectively.

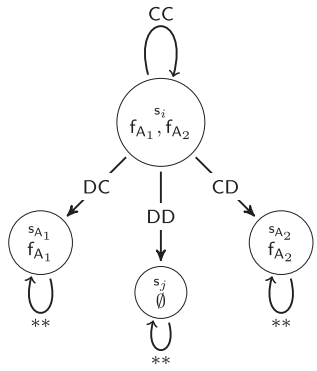


Fig. 2. The CGS $\mathcal{G}_{PD}$.

There are four states in $\mathrm{St} \triangleq \{s_i, s_{A_1}, s_{A_2}, s_j\}$. In the idle state s_i the agents are waiting for the interrogation, while s_j represents the jail for both of them. Instead the remaining states s_{A_1} and s_{A_2} indicate the situations in which only one of the agents becomes definitely free. To characterize the different meaning of these states, we use the atomic propositions in $\mathrm{AP} \triangleq \{f_{A_1}, f_{A_2}\}$, which denote who is "free," by defining the following labeling: $\lambda(s_i) \triangleq \{f_{A_1}, f_{A_2}\}$, $\lambda(s_{A_1}) \triangleq \{f_{A_1}\}$, $\lambda(s_{A_2}) \triangleq \{f_{A_2}\}$, and $\lambda(s_j) \triangleq \emptyset$. The transition function τ can be easily deduced by the figure.

As another example in which it is possible to discuss about the ability of an agent to be safe with respect to the behavior of malicious agents, we consider a generalization of the well-known robber-versuss-cops game (see Aigner and Fromme [1984] for a survey).

Example 2.4 (*Romeo and Juliet Game*). In the *Romeo and Juliet* game, Romeo, being able to move inside Juliet's house, aims to reach her balcony, trying to avoid meeting both the Montague and Capulet families, who are able to move concurrently with him in the house. Graphically, the house is divided in rooms, and each room has doors on all internal walls. The topology of the house depends on the status of these doors that can be closed or open.

Additionally, Romeo is helped by the accomplice Shakespeare who controls the doors of the rooms. Specifically, at each step of the game, Shakespeare can switch the status of at most two doors at a time (i.e., from open to closed or vice versa). Hence, he can modify the shape of the house during a play. We consider Romeo meeting the families in two cases: if they are in the same room in a certain phase of the play (Romeo is captured) or if they are in adjacent rooms with the connecting door open (Romeo is shot). In each step, the agents Romeo, the Montagues family, and the Capulet family can independently decide to stay in the room or, in the case a connection door to another room is open, move to this one. A sample game given in Figure 3 is made by nine rooms, with Room 9 being the balcony, Romeo in Room 1, the Montague family in Room 3, and the Capulet family in Room 7, respectively, and all the doors open as initial state of the game. We denote with $\mathrm{R} = [1, 9]$ the set of rooms and $\mathrm{D} = \{d = (r_1, r_2) \in \mathrm{R} \times \mathrm{R} : r_1 \text{ is adjacent to } r_2\}$ the set of doors.

We can model this situation with a CGS $\mathcal{G}_{RJ} = \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}, \mathrm{St}, \lambda, \tau, s_0 \rangle$ with $\mathrm{AP} \triangleq \{\mathsf{c}, \mathsf{r}\}$ standing for *Captured* or *Shot* (c) and *Reached* (r); $\mathrm{Ag} \triangleq \{\mathsf{R}, \mathsf{M}, \mathsf{C}, \mathsf{S}\}$, standing for *Romeo* ($\mathsf{R}$), *Montagues* ($\mathsf{M}$), *Capulets* ($\mathsf{C}$), *Shakespeare* ($\mathsf{S}$), respectively; $\mathrm{Ac} \triangleq \mathrm{M} \cup \mathrm{C}$, where $\mathrm{M} \triangleq \{\mathsf{stay}, \mathsf{up}, \mathsf{down}, \mathsf{left}, \mathsf{right}\}$ is the set of moves for Romeo and families, and $\mathrm{C} = \{d \in 2^{\mathrm{D}} : |d| \leq 2\}$ is the set of possible switching of doors for Shakespeare, in which the empty set stands for no switching; $\mathrm{St} \triangleq \mathrm{R} \times \mathrm{R}^2 \times \{0, 1\}^{\mathrm{D}}$ be the set of states

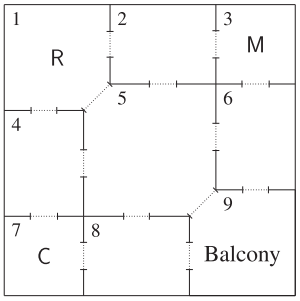


Fig. 3. Romeo and Juliet game.

in which the first coordinate stands for the position of Romeo, the second and third coordinates stand for the positions of the Montagues and Capulets, respectively, and the fourth coordinate is a function describing the status open or closed of each door in the house. The initial state is given by $(1, 3, 7, \mathsf{f}_0)$, where $\mathsf{f}_0 : \mathrm{D} \to \{0, 1\}$ is the function constant to 0 (i.e., all the doors are open). The labeling function is given as follows: for each state $s = (n, m_1, m_2, \mathsf{f}) \in \mathrm{St}$, $\mathsf{r} \in \lambda(s)$ if and only if $n = 9$,[3] $\mathsf{c} \in \lambda(s)$ if and only if there exists $i \in \{1, 2\}$ such that either $n = m_i$ or $(n, m_i) \in \mathrm{D}$ and $\mathsf{f}((n, m_i)) = 0$.[4] The transition function can be easily deduced by the figure. For the sake of simplicity, all the states labeled with r only have a loop in the game.[5]

2.2. Syntax

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

Definition 2.5 (SL. *Syntax*). SL formulas are built inductively from the sets of atomic propositions AP, variables Var, and agents Ag, by using the following grammar, where $p \in \mathrm{AP}$, $x \in \mathrm{Var}$, and $a \in \mathrm{Ag}$:

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

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

Observe that, by construction, LTL is a proper syntactic fragment of SL (i.e., $\mathrm{LTL} \subset \mathrm{SL}$). 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 metavariables 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 $\mathsf{sub}(\varphi)$, we formally denote the set of subformulas of an SL formula φ. For instance, consider $\varphi = \langle\langle \mathsf{x} \rangle\rangle(\alpha, \mathsf{x})(\mathsf{F}\,\mathsf{p})$. Then, it is immediate to see that $\mathsf{sub}(\varphi) = \{\varphi, (\alpha, \mathsf{x})(\mathsf{F}\,\mathsf{p}), (\mathsf{F}\,\mathsf{p}), \mathsf{p}, \mathsf{t}\}$.

[3]Romeo has reached the balcony.

[4]Romeo is in the same room with a family or they are close with the door open.

[5]For the sake of simplicity, we assume that each agent, at each state of the game, has a specific subset of possible choices of actions. For instance, if Romeo stays in Room 4 and the door (4, 5) is closed, then only the actions up, down, and stay are available to him. It is easy to see that such assumption can be overcame in a suitable, but even tricky, CGS that is somehow equivalent.

Normally, predicative logics need the concepts of free and bound placeholders to formally define the meaning of their formulas. The placeholders are used to represent particular positions in syntactic expressions that have to be highlighted because 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 placeholders, as it can be noted by its syntax, to distinguish between the quantification of a strategy and its application by an agent. Consequently, we need a way to differentiate whether an agent has an associated strategy via a variable and whether 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 2.6 (SL *Free Agents / Variables*). The set of free agents/variables of an SL formula is given by the function $\mathsf{free} : \mathrm{SL} \to 2^{\mathrm{Ag} \cup \mathrm{Var}}$ defined as follows:

(i) $\mathsf{free}(p) \triangleq \emptyset$, where $p \in \mathrm{AP}$;
(ii) $\mathsf{free}(\neg\varphi) \triangleq \mathsf{free}(\varphi)$;
(iii) $\mathsf{free}(\varphi_1 \mathsf{Op}\, \varphi_2) \triangleq \mathsf{free}(\varphi_1) \cup \mathsf{free}(\varphi_2)$, where $\mathsf{Op} \in \{\wedge, \vee\}$;
(iv) $\mathsf{free}(\mathsf{X}\, \varphi) \triangleq \mathrm{Ag} \cup \mathsf{free}(\varphi)$;
(v) $\mathsf{free}(\varphi_1 \mathsf{Op}\, \varphi_2) \triangleq \mathrm{Ag} \cup \mathsf{free}(\varphi_1) \cup \mathsf{free}(\varphi_2)$, where $\mathsf{Op} \in \{\mathsf{U}, \mathsf{R}\}$;
(vi) $\mathsf{free}(\mathsf{Qn}\, \varphi) \triangleq \mathsf{free}(\varphi) \setminus \{x\}$, where $\mathsf{Qn} \in \{\langle\langle x \rangle\rangle, [[x]] : x \in \mathrm{Var}\}$;
(vii) $\mathsf{free}((a, x)\varphi) \triangleq \mathsf{free}(\varphi)$, if $a \notin \mathsf{free}(\varphi)$, where $a \in \mathrm{Ag}$ and $x \in \mathrm{Var}$;
(viii) $\mathsf{free}((a, x)\varphi) \triangleq (\mathsf{free}(\varphi) \setminus \{a\}) \cup \{x\}$, if $a \in \mathsf{free}(\varphi)$, where $a \in \mathrm{Ag}$ and $x \in \mathrm{Var}$.

A formula φ without free agents (variables, respectively), that is, with $\mathsf{free}(\varphi) \cap \mathrm{Ag} = \emptyset$ ($\mathsf{free}(\varphi) \cap \mathrm{Var} = \emptyset$, respectively), is named *agent-closed* (*variable-closed*, respectively). If φ is both agent- and variable-closed, it is referred to as a *sentence*. The function $\mathsf{snt} : \mathrm{SL} \to 2^{\mathrm{SL}}$ returns the set of subsentences $\mathsf{snt}(\varphi) \triangleq \{\phi \in \mathsf{sub}(\varphi) : \mathsf{free}(\phi) = \emptyset\}$, 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 $\mathsf{free}(\varphi) = \{\gamma, \mathsf{y}\}$, 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 earlier. Then, we have $\mathsf{free}((\alpha, \mathsf{z})\varphi) = \mathsf{free}(\varphi)$ and $\mathsf{free}((\gamma, \mathsf{z})\varphi) = \{\mathsf{y}, \mathsf{z}\}$, since α is not free in φ but γ is, i.e., $\alpha \notin \mathsf{free}(\varphi)$ and $\gamma \in \mathsf{free}(\varphi)$. 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 the one of *alternation number* of quantifiers, that is, 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 binds 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) are 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 because a higher alternation can usually means a higher complexity. By $\mathsf{alt}(\varphi)$, we formally denote the alternation number of an SL formula φ. Furthermore, the fragment SL[k-alt] $\triangleq \{\varphi \in \mathrm{SL} : \forall \varphi' \in \mathsf{sub}(\varphi).\, \mathsf{alt}(\varphi') \leq k\}$ 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 $\mathsf{alt}(\varphi)$ is 1 and not 3, as one can think at a first glance, because φ' is a sentence. Moreover, it holds that $\mathsf{alt}(\varphi') = 1$. Hence, $\varphi \in \text{SL}[1\text{-alt}]$. On the other hand, if we substitute φ' with $\varphi'' = [[\mathsf{x}]](\alpha,\mathsf{x})(\mathsf{X}\,\mathsf{p})$, we have that $\mathsf{alt}(\varphi) = 2$ because φ'' is not a sentence. Thus, in the latter case, it holds that $\varphi \notin \text{SL}[1\text{-alt}]$ but $\varphi \in \text{SL}[2\text{-alt}]$.

At this point, 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 article. The first concept we introduce is the well-known deterministic concurrent multiplayer *Nash equilibrium* for Boolean valued payoffs.

Example 2.7 (*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 $(\mathsf{x}_1,\ldots,\mathsf{x}_n)$ that is a Nash equilibrium (NE) for $\alpha_1,\ldots,\alpha_n$ with respect to $\psi_1,\ldots,\psi_n$ by using the SL[1-alt] sentence $\varphi_{NE} \triangleq \langle\langle\mathsf{x}_1\rangle\rangle(\alpha_1,\mathsf{x}_1)\cdots\langle\langle\mathsf{x}_n\rangle\rangle(\alpha_n,\mathsf{x}_n)\,\psi_{NE}$, where $\psi_{NE} \triangleq \bigwedge_{i=1}^{n}(\langle\langle\mathsf{y}\rangle\rangle(\alpha_i,\mathsf{y})\psi_i) \rightarrow \psi_i$ is a variable-closed formula. Informally, this asserts that every agent α_i has x_i as one of the best strategy with respect to 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.

Note that the syntactic feature of SL that allows us to represent Nash equilibria is the binding construct. Indeed, by means of a suitable usage of it, we can compare, in a Boolean way, the outcomes of two strategy profiles in which only one agent has changed his choice. This cannot be done in ATL*, since the use of an agent modality necessarily requantify the strategies associated with all the agents. Therefore, we cannot change in ATL* just a strategy of a single selected agent.

In game theory, it is important to recall that an equilibrium is not always stable. Indeed, there are games like the PD of Example having Nash equilibria that are instable. One of the simplest concepts of stability that is possible to think is called the *stability profile*.

Example 2.8 (*Stability Profile*). Think about the same situation of the previous example on NE. Then, a stability profile (SP) is a strategy profile $(\mathsf{x}_1,\ldots,\mathsf{x}_n)$ for $\alpha_1,\ldots,\alpha_n$ with respect to $\psi_1,\ldots,\psi_n$ 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 $\varphi_{SP} \triangleq \langle\langle\mathsf{x}_1\rangle\rangle(\alpha_1,\mathsf{x}_1)\cdots\langle\langle\mathsf{x}_n\rangle\rangle(\alpha_n,\mathsf{x}_n)\,\psi_{SP}$, where $\psi_{SP} \triangleq \bigwedge_{i,j=1,i\neq j}^{n}\psi_j \rightarrow [[\mathsf{y}]]((\psi_i \leftrightarrow (\alpha_i,\mathsf{y})\psi_i) \rightarrow (\alpha_i,\mathsf{y})\psi_j)$. 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 $\varphi_{SNE} \triangleq \langle\langle\mathsf{x}_1\rangle\rangle(\alpha_1,\mathsf{x}_1)\cdots\langle\langle\mathsf{x}_n\rangle\rangle(\alpha_n,\mathsf{x}_n)\,\psi_{SP} \wedge \psi_{NE}$.

2.3. Basic Concepts

Before continuing with the description of our logic, we have to introduce some basic concepts, regarding a generic CGS, which are at the base of the semantics formalization. Most of these notions are inspired from the classical ones given in the settings of game structures and graphs. However, as there are different formalizations of them in the literature, as well as for the sake of completeness, we prefer to give them here in full

detail. We recall that a description of the used mathematical notation is reported in the accompanying electronic appendix.

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. A track (path, respectively) in a CGS $\mathcal{G}$ is a finite (an infinite, respectively) sequence of states $\rho \in \mathrm{St}^*$ ($\pi \in \mathrm{St}^\omega$, respectively) such that for all $i \in [0, |\rho|-1[$ ($i \in \mathbb{N}$, respectively), there exists a decision $\mathsf{d} \in \mathrm{Dc}$ such that $(\rho)_{i+1} = \tau((\rho)_i, \mathsf{d})$ ($(\pi)_{i+1} = \tau((\pi)_i, \mathsf{d})$, respectively).[6] A track ρ is nontrivial if it has nonzero length (i.e., $|\rho| > 0$ that is $\rho \neq \varepsilon$).[7] The set $\mathrm{Trk} \subseteq \mathrm{St}^+$ ($\mathrm{Pth} \subseteq \mathrm{St}^\omega$, respectively) contains all nontrivial tracks (paths, respectively). Moreover, $\mathrm{Trk}(s) \triangleq \{\rho \in \mathrm{Trk} : \mathsf{fst}(\rho) = s\}$ ($\mathrm{Pth}(s) \triangleq \{\pi \in \mathrm{Pth} : \mathsf{fst}(\pi) = s\}$, respectively) indicates the subsets of tracks (paths, respectively) starting at a state $s \in \mathrm{St}$.[8] For instance, consider the PRS game of Example 2.2. Then, $\rho = \mathsf{s}_i \cdot \mathsf{s}_\mathsf{A} \in \mathrm{St}^+$ and $\pi = \mathsf{s}_i{}^\omega \in \mathrm{St}^\omega$ are, respectively, a track and a path in the CGS $\mathcal{G}_{PRS}$. Moreover, it holds that $\mathrm{Trk} = \mathsf{s}_i{}^+ + \mathsf{s}_i{}^* \cdot (\mathsf{s}_\mathsf{A}{}^+ + \mathsf{s}_\mathsf{B}{}^+)$ and $\mathrm{Pth} = \mathsf{s}_i{}^\omega + \mathsf{s}_i{}^* \cdot (\mathsf{s}_\mathsf{A}{}^\omega + \mathsf{s}_\mathsf{B}{}^\omega)$.

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 because the same strategy can be used by more than one agent at the same time. A strategy in a CGS $\mathcal{G}$ is a partial function $\mathsf{f} : \mathrm{Trk} \rightharpoonup \mathrm{Ac}$ that maps each nontrivial 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, that is, $\mathsf{dom}(\mathsf{f}) = \mathrm{Trk}(s)$. The set $\mathrm{Str} \triangleq \mathrm{Trk} \rightharpoonup \mathrm{Ac}$ ($\mathrm{Str}(s) \triangleq \mathrm{Trk}(s) \to \mathrm{Ac}$, respectively) contains all (s-total, respectively) strategies. An example of strategy in the CGS $\mathcal{G}_{PRS}$ is the function $\mathsf{f}_1 \in \mathrm{Str}(\mathsf{s}_i)$ 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 $\mathsf{f}_2 \in \mathrm{Str}(\mathsf{s}_i)$ 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.

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 2.9 (*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, that is, $\mathrm{Ag} \subseteq \mathsf{dom}(\chi)$. For a state $s \in \mathrm{St}$, it is said that χ is s-total if all strategies $\chi(l)$ are s-total, for $l \in \mathsf{dom}(\chi)$. The set $\mathrm{Asg} \triangleq \mathrm{Var} \cup \mathrm{Ag} \rightharpoonup \mathrm{Str}$ ($\mathrm{Asg}(s) \triangleq \mathrm{Var} \cup \mathrm{Ag} \rightharpoonup \mathrm{Str}(s)$, respectively) contains all (s-total, respectively) assignments. Moreover, $\mathrm{Asg}(\mathrm{X}) \triangleq \mathrm{X} \to \mathrm{Str}$ ($\mathrm{Asg}(\mathrm{X}, s) \triangleq \mathrm{X} \to \mathrm{Str}(s)$, respectively) indicates the subset of X-*defined* (s-total, respectively) assignments, that is, (s-total, respectively) 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 $\{\mathsf{A}, \mathsf{x}\}$, whose values are f_1 on A and f_2 on x, where the strategies

[6]The notation $(w)_i \in \Sigma$ indicates the element of index $i \in [0, |w|[$ of a nonempty sequence $w \in \Sigma^\infty$.
[7]The Greek letter ε stands for the empty sequence.
[8]By $\mathsf{fst}(w) \triangleq (w)_0$, it is denoted the first elementof a nonempty sequence $w \in \Sigma^\infty$.

$\mathsf{f}_1, \mathsf{f}_2 \in \mathrm{Str}(\mathsf{s}_i)$ are those described earlier. Another examples is given by the assignment $\chi_2 \in \mathrm{Asg}$, defined on the set $\{\mathsf{A}, \mathsf{B}\}$, such that $\chi_2(\mathsf{A}) = \chi_1(\mathsf{x})$ and $\chi_2(\mathsf{B}) = \chi_1(\mathsf{A})$. Note that both assignments are s_i-total and the latter is also complete while the former is not.

Given an assignment $\chi \in \mathrm{Asg}$, an agent or variable $l \in \mathrm{Var} \cap \mathrm{Ag}$, and a strategy $\mathsf{f} \in \mathrm{Str}$, it is important to define a notation to represent the redefinition of χ, that is, a new assignment equal to the first on all elements of its domain but l, on which it assumes the value f. Then, formally, with $\chi[l \mapsto \mathsf{f}] \in \mathrm{Asg}$, we denote 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, that is, $\chi[l \mapsto \mathsf{f}](l) \triangleq \mathsf{f}$ 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 aforementioned notation. It is worth observing that if χ and f are s-total, then $\chi[l \mapsto \mathsf{f}]$ is s-total, as well.

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 with it.

Definition 2.10 (*Plays*). A path $\pi \in \mathrm{Pth}(s)$ starting at a state $s \in \mathrm{St}$ is a *play* with respect a complete s-total assignment $\chi \in \mathrm{Asg}(s)$ ((χ, s)-*play*) 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 $a \in \mathrm{Ag}$.[9] 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, at a certain step of the play, is used to calculate the current state and its updated assignment.

Definition 2.11 (*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 $\pi = \mathsf{play}(\chi, s)$, and by $(\chi)_{(\pi)_{\leq i}}$, we are denoting the $(\pi)_i$-total assignment, with $\mathsf{dom}((\chi)_{(\pi)_{\leq i}}) = \mathsf{dom}(\chi)$, such that for all $l \in \mathsf{dom}(\chi)$, $\mathsf{dom}((\chi)_{(\pi)_{\leq i}}(l)) = \{\rho \in \mathrm{Trk}((\pi)_i) : (\pi)_{\leq i} \cdot \rho \in \mathsf{dom}(\chi(l))\}$ and $(\chi)_{(\pi)_{\leq i}}(l)(\rho) = \chi(l)((\pi)_{\leq i} \cdot \rho)$ for all $\rho \in \mathsf{dom}((\chi)_{(\pi)_{\leq i}}(l))$.

To avoid any ambiguity of interpretation of the described notions, we sometimes 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 with respect to concurrent game structures. For an SL formula φ, 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 are 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).

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

Definition 2.12 (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:
   (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 $x \in \mathrm{Var}$ 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 $\mathsf{f} \in \mathrm{Str}(s)$ 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 $\mathsf{f} \in \mathrm{Str}(s)$ it holds that $\mathcal{G}, \chi[x \mapsto \mathsf{f}], s \models \varphi$.
(4) For an agent $a \in \mathrm{Ag}$, a variable $x \in \mathrm{Var}$, 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 $k \leq i$ such that $\mathcal{G}, (\chi, s)^i \models \varphi_2$ and, for all indexes $j \in \mathbb{N}$ with $k \leq j < i$, 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 $k \leq i$, it holds that $\mathcal{G}, (\chi, s)^i \models \varphi_2$ or there is an index $j \in \mathbb{N}$ with $k \leq j < i$ 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.

To complete the description of the semantics, we now give the classic notions of *model* and *satisfiability* of an SL sentence. We say that a CGS $\mathcal{G}$ is a model of an SL sentence φ, in symbols $\mathcal{G} \models \varphi$, if $\mathcal{G}, \varnothing, s_0 \models \varphi$.[10] In general, we also say that $\mathcal{G}$ is a model for φ on $s \in \mathrm{St}$, 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. Given two SL formulas φ_1 and φ_2, with $\mathsf{free}(\varphi_1) = \mathsf{free}(\varphi_2)$, we say that $\varphi_1$implies φ_2, in symbols $\varphi_1 \Rightarrow \varphi_2$, if for all CGSs $\mathcal{G}$, states $s \in \mathrm{St}$, and $\mathsf{free}(\varphi_1)$-defined s-total assignments $\chi \in \mathrm{Asg}(\mathsf{free}(\varphi_1), s)$, 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 this article, especially when we describe a decision procedure, we may consider formulas in *existential normal form* (*enf*) and *positive normal form* (*pnf*), that is, formulas in which only existential quantifiers appear or in which the negation is applied only to atomic propositions, respectively. 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 $\neg(a, x)\varphi \equiv (a, x)\neg\varphi$.

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

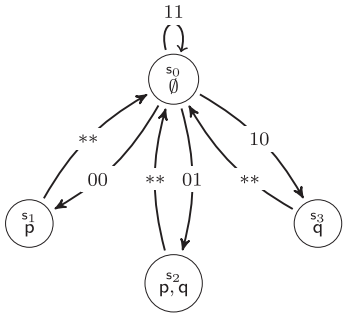


Fig. 4. The CGS $\mathcal{G}_{SV}$.

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

Example 2.13 (*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 $\mathsf{X}\,\mathsf{p}$ and $\mathsf{X}\,\mathsf{q}$. A model for φ is given by the CGS $\mathcal{G}_{SV} \triangleq \langle \{\mathsf{p}, \mathsf{q}\}, \{\alpha, \beta\}, \{0, 1\}, \{\mathsf{s}_0, \mathsf{s}_1, \mathsf{s}_2, \mathsf{s}_3\}, \lambda, \tau, \mathsf{s}_0 \rangle$, where $\lambda(\mathsf{s}_0) \triangleq \emptyset$, $\lambda(\mathsf{s}_1) \triangleq \{\mathsf{p}\}$, $\lambda(\mathsf{s}_2) \triangleq \{\mathsf{p}, \mathsf{q}\}$, $\lambda(\mathsf{s}_3) \triangleq \{\mathsf{q}\}$, $\tau(\mathsf{s}_0, (0, 0)) \triangleq \mathsf{s}_1$, $\tau(\mathsf{s}_0, (0, 1)) \triangleq \mathsf{s}_2$, $\tau(\mathsf{s}_0, (1, 0)) \triangleq \mathsf{s}_3$, and all the remaining transitions (with any decision) go to s_0. In Figure 4, 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 $(\alpha, \mathsf{x})(\beta, \mathsf{y})(\mathsf{X}\,\mathsf{p})$ 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 $(\alpha, \mathsf{y})(\beta, \mathsf{z})(\mathsf{X}\,\mathsf{q})$ 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 2.14 (*Equilibrium Profiles*). Let us first consider the CGS $\mathcal{G}_{PD}$ of the prisoner's dilemma described in the Example 2.3. Intuitively, each of the two accomplices A_1 and A_2 want to avoid the prison. These goals can be represented by the LTL formulas $\psi_{\mathsf{A}_1} \triangleq \mathsf{G}\,\mathsf{f}_{\mathsf{A}_1}$ and $\psi_{\mathsf{A}_2} \triangleq \mathsf{G}\,\mathsf{f}_{\mathsf{A}_2}$, respectively. The existence of a Nash equilibrium in $\mathcal{G}_{PD}$ for the two accomplices with respect to the aforementioned goals can be written as $\phi_{NE} \triangleq \langle\langle \mathsf{x}_1 \rangle\rangle(\mathsf{A}_1, \mathsf{x}_1)\langle\langle \mathsf{x}_2 \rangle\rangle(\mathsf{A}_2, \mathsf{x}_2)\ \psi_{NE}$, where $\psi_{NE} \triangleq ((\langle\langle \mathsf{y} \rangle\rangle(\mathsf{A}_1, \mathsf{y})\psi_{\mathsf{A}_1}) \rightarrow \psi_{\mathsf{A}_1}) \wedge ((\langle\langle \mathsf{y} \rangle\rangle(\mathsf{A}_2, \mathsf{y})\psi_{\mathsf{A}_2}) \rightarrow \psi_{\mathsf{A}_2})$, which results to be an instantiation of the general sentence φ_{NE} of Example 2.7. In the same way, the existence of a stable Nash equilibrium can be represented with the sentence $\phi_{SNE} \triangleq \langle\langle \mathsf{x}_1 \rangle\rangle(\mathsf{A}_1, \mathsf{x}_1)\langle\langle \mathsf{x}_2 \rangle\rangle(\mathsf{A}_2, \mathsf{x}_2)\ \psi_{NE} \wedge \psi_{SP}$, where $\psi_{SP} \triangleq (\psi_1 \rightarrow [\![\mathsf{y}]\!]((\psi_2 \leftrightarrow (\mathsf{A}_2, \mathsf{y})\psi_2) \rightarrow (\mathsf{A}_2, \mathsf{y})\psi_1)) \wedge (\psi_2 \rightarrow [\![\mathsf{y}]\!]((\psi_1 \leftrightarrow (\mathsf{A}_1, \mathsf{y})\psi_1) \rightarrow (\mathsf{A}_1, \mathsf{y})\psi_2))$, which is a particular case of the sentence φ_{SNE} of Example 2.8. Now, it is easy to see that $\mathcal{G}_{PD} \models \phi_{SNE}$ and, so, $\mathcal{G}_{PD} \models \phi_{NE}$. Indeed, an assignment $\chi \in \mathrm{Asg}_{\mathcal{G}_{PD}}(\mathrm{Ag}, \mathsf{s}_i)$, for which $\chi(\mathsf{A}_1)(\mathsf{s}_i) = \chi(\mathsf{A}_2)(\mathsf{s}_i) = \mathsf{D}$, is a stable equilibrium profile i.e., it is such that $\mathcal{G}_{PD}, \chi, \mathsf{s}_i \models \psi_{NE} \wedge \psi_{SP}$. This is due to the fact that if an agent A_k, for $k \in \{1, 2\}$, chooses another strategy $\mathsf{f} \in \mathrm{Str}_{\mathcal{G}_{PD}}(\mathsf{s}_i)$, he is still unable to achieve his goal ψ_k, i.e., $\mathcal{G}_{PD}, \chi[\mathsf{A}_k \mapsto \mathsf{f}], \mathsf{s}_i \not\models \psi_k$, 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

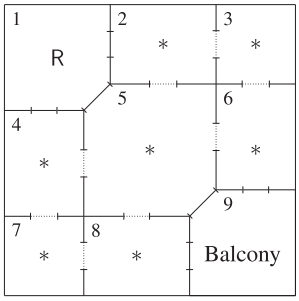


Fig. 5. Winning position.

equilibria too. One of these is represented by the assignment $\chi' \in \mathrm{Asg}_{\mathcal{G}_{PD}}(\mathrm{Ag}, \mathsf{s}_i)$, for which $\chi'(\mathsf{A}_1)(\mathsf{s}_i{}^j) = \chi'(\mathsf{A}_2)(\mathsf{s}_i{}^j) = \mathsf{C}$ for all $j \in \mathbb{N}$. Indeed, we have that $\mathcal{G}_{PD}, \chi', \mathsf{s}_i \models \psi_{NE}$, since $\mathcal{G}_{PD}, \chi', \mathsf{s}_i \models \psi_1$ and $\mathcal{G}_{PD}, \chi', \mathsf{s}_i \models \psi_2$, but $\mathcal{G}_{PD}, \chi', \mathsf{s}_i \not\models \psi_{SP}$. The latter property holds because, if one of the agents A_k, for $k \in \{1, 2\}$, chooses a different strategy $\mathsf{f}' \in \mathrm{Str}_{\mathcal{G}_{PD}}(\mathsf{s}_i)$ for which there is a $j \in \mathbb{N}$ such that $\mathsf{f}'(\mathsf{s}_i{}^j) = \mathsf{D}$, he cannot improve his payoff but makes surely worse the payoff of the other agent, that is, $\mathcal{G}_{PD}, \chi'[\mathsf{A}_k \mapsto \mathsf{f}'], \mathsf{s}_i \models \psi_k$ but $\mathcal{G}_{PD}, \chi'[\mathsf{A}_k \mapsto \mathsf{f}'], \mathsf{s}_i \not\models \psi_{3-k}$. Finally, consider the CGS $\mathcal{G}_{PRS}$ of the paper, rock, and scissor game described in the Example 2.2 together with the associated formula for the Nash equilibrium $\phi_{NE} \triangleq \langle\!\langle \mathsf{x}_1 \rangle\!\rangle(\mathsf{A}, \mathsf{x}_1)\langle\!\langle \mathsf{x}_2 \rangle\!\rangle(\mathsf{B}, \mathsf{x}_2)\, \psi_{NE}$, where $\psi_{NE} \triangleq ((\langle\!\langle \mathsf{y} \rangle\!\rangle(\mathsf{A}, \mathsf{y})\psi_{\mathsf{A}}) \rightarrow \psi_{\mathsf{A}}) \wedge ((\langle\!\langle \mathsf{y} \rangle\!\rangle(\mathsf{B}, \mathsf{y})\psi_{\mathsf{B}}) \rightarrow \psi_{\mathsf{B}})$ with $\psi_{\mathsf{A}} \triangleq \mathsf{F}\, \mathsf{w}_{\mathsf{A}}$ and $\psi_{\mathsf{B}} \triangleq \mathsf{F}\, \mathsf{w}_{\mathsf{B}}$ representing the LTL temporal goals for Alice and Bob, respectively. Then, it is not hard to see that $\mathcal{G}_{PRS} \not\models \phi_{NE}$, that is, 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.

In the next example, we show the expressive power of alternation of quantifications in SL by pointing out an example based on the "Romeo and Juliet" game, shown in Example 2.4.

Example 2.15 (*Quantification modalities*). First consider the CGS $\mathcal{G}_{RJ}$ described in Example 2.4. Romeo R, helped by Shakespeare S, wants to reach the balcony of his loved Juliet while never meeting the Montague family M or the Capulet family C. According to the CGS, this aim can be easily expressed with the LTL formula $\psi = \mathsf{F}\, \mathsf{r} \wedge \mathsf{G}\, \neg \mathsf{c}$. The game can be played with several quantification modalities, each of them having a different meaning. For example, we can consider the formula $\varphi_1 = [\![\mathsf{z}]\!][\![\mathsf{w}]\!]\langle\!\langle x \rangle\!\rangle\langle\!\langle \mathsf{y} \rangle\!\rangle\flat\psi$, where $\flat = (\mathsf{R}, \mathsf{x})(\mathsf{S}, \mathsf{y})(\mathsf{M}, \mathsf{z})(\mathsf{C}, \mathsf{w})$. This means that both R and S have full visibility about the moves of the families in the house. Conversely, we can consider the formula $\varphi_2 = \langle\!\langle x \rangle\!\rangle\langle\!\langle \mathsf{y} \rangle\!\rangle[\![\mathsf{z}]\!][\![\mathsf{w}]\!]\flat\psi$, which means that both R and S have to play a strategy that is winning no matter what the families do. Moreover, one can select a modality in which the alternation of the quantification is greater than one. For instance, in the formula $\varphi_3 = \langle\!\langle x \rangle\!\rangle[\![\mathsf{z}]\!][\![\mathsf{w}]\!]\langle\!\langle \mathsf{y} \rangle\!\rangle\flat\psi$, Romeo has to play independently on the families, whereas Shakespeare can select a strategy according to what the families are doing in the house. In other words, a team of agents has visibility on the adversary team that differs agent by agent, and depends on the order in which their strategies are quantified in the formula. It is worth noting that, while φ_1 and φ_2 can be represented also in ATL*, φ_3 does not have an equivalent ATL* formula, due to its alternation number of quantifications equal to 2.

It is easy to see that the position in Figure 5, with rooms 1 and 9 completely isolated and families standing in the remaining sector of the house composed by rooms from 2 to 8, is winning for Romeo and Shakespeare. In fact, a trivial strategy for them is

given by Romeo remaining in room 1 until Shakespeare closes the doors needed to block the families in two rooms (possibly the same) of the house, then opening a path toward the balcony for Romeo. Note that such a path always exists because there are three different nonintersecting paths from room 1 to room 9 (i.e., $1-2-3-6-9$, $1-4-7-8-9$, and $1-5-9$) and the families can occupy at most two of them.

Moreover, it is easy to see that the winning position in Figure 5 cannot be reached if Shakespeare does not have full visibility of the families' moves. Indeed, starting from the initial position, Shakespeare is forced to close doors $(1,2)$ and $(1,4)$ to prevent Romeo to be shot at the second step because the Montagues and Capulets can move in 2 and 4, respectively. But, by closing those doors, the Montagues (the Capulets, respectively) can move in 6 (8, respectively), being able to reach the balcony room in the next step of the game and holding there indefinitely. This makes it impossible to eventually satisfy r without satisfying c, as well. Hence, we have that $\mathcal{G}_{RJ} \not\models \varphi_2$.

On the other hand, if Shakespeare knows where the families are moving from the initial state, then he can close the nearest doors and prevent them to reach the key rooms (i.e., 1 and 9) in the next steps. Indeed, if the Montagues and Capulets are going to move in 2 and 4, then Shakespeare closes the doors $(1,2)$ and $(1,4)$; otherwise, if the Montagues and Capulets are going to move in 6 and 8, then Shakespeare closes doors $(6,9)$ and $(7,9)$. Again, the "mixed" strategies of the Montagues and Capulets given by moving in 2 and 8 or 4 and 6 can be neutralized by a suitable choice for Shakespeare. Moreover, it is easy to see that, after a well-played initial step, the the winning position of Figure 5 is easily reachable by Romeo and Shakespeare, as the latter can close doors $(1,5)$ and $(5,9)$ at the second step and then close the remaining doors to isolate rooms 1 and 9.

Thanks to this reasoning, we have that $\mathcal{G}_{RJ} \models \varphi_3$.

Finally, we discuss another examples that is useful to point out the power of forcing two or more agents to share a strategy. Hex is a two-player game, red versus blue, in which each player in turn places a stone of his color on a single empty hexagonal cell of the rhomboidal playing board having opposite sides equally colored, either red or blue. The goal of each player is to be the first to form a path connecting the opposing sides of the board marked by his color. It is easy to prove that the stealing-strategy argument does not lead to a winning strategy in Hex, that is, if the player that moves second copies the moves of the opponent, he surely loses the play. It is possible to formalize this fact in SL as in the following example.

Example 2.16 (*Stealing-Strategy argument in Hex*). First, model Hex with a CGS $\mathcal{G}_H$ whose states represent every possible configuration reached during a play between Player "r" red and "b" blue. Then, verify the negation of the stealing-strategy argument by checking $\mathcal{G}_H \models \langle\!\langle \mathsf{x} \rangle\!\rangle (\mathsf{r}, \mathsf{x})(\mathsf{b}, \mathsf{x})(\mathsf{F}\, \mathsf{cnc}_\mathsf{r})$. Intuitively, this sentence says that agent r has a strategy that, once it is copied (bound) by b it allows the former to win, that is, to be the first to connect the related red edges ($\mathsf{F}\, \mathsf{cnc}_\mathsf{r}$).

It should be noted that the syntax of SL does not allow to express manipulation of strategies. For example, in the Hex game, for Player b, we cannot express the effects of a strategy obtained by a manipulation of the one chosen by Player r. As a further example, consider a CGS having $\mathrm{Ac} = \{0, 1\}$. Then, in SL, it is not possible to describe a comparison between the outcomes that a player may obtain by replacing one of his strategies f with the associated flipping f', that is, $\mathsf{f}'(\rho) = 1 - \mathsf{f}(\rho)$ for all $\rho \in \mathsf{dom}(\mathsf{f})$.

3. MODEL-CHECKING HARDNESS

In this section, we show the nonelementary 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 Section 3.1, we first recall syntax and semantics of quantified propositional temporal logic [Sistla 1983]. Then, in Section 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) 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 $p \in \mathrm{AP}$:

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

QPtl denotes the infinite set of formulas generated by the aforementioned 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 $k \in \mathbb{N}$.

To define the semantics of QPtl, we first have to introduce the concepts of truth evaluations used to interpret the meaning of atomic propositions at the passing of time. A *temporal truth evaluation* is a function $\mathsf{tte} : \mathbb{N} \to \{\mathfrak{f}, \mathfrak{t}\}$ that maps each natural number to a Boolean value. Moreover, a *propositional truth evaluation* is a partial function $\mathsf{pte} : \mathrm{AP} \rightharpoonup \mathrm{TTE}$ mapping every atomic proposition in its domain to a temporal truth evaluation. The sets $\mathrm{TTE} \triangleq \mathbb{N} \to \{\mathfrak{f}, \mathfrak{t}\}$ and $\mathrm{PTE} \triangleq \mathrm{AP} \rightharpoonup \mathrm{TTE}$ 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 $\mathsf{free}(\varphi) \subseteq \mathsf{dom}(\mathsf{pte})$ and a number k, we write $\mathsf{pte}, k \models \varphi$ to indicate that the formula φ holds at the k-th position of the pte.

Definition 3.2 (QPtl *Semantics*). For all QPtl formulas φ, propositional truth evaluation $\mathsf{pte} \in \mathrm{PTE}$ with $\mathsf{free}(\varphi) \subseteq \mathsf{dom}(\mathsf{pte})$, and numbers $k \in \mathbb{N}$, the modeling relation $\mathsf{pte}, k \models \varphi$ is inductively defined as follows:

(1) $\mathsf{pte}, k \models p$ if and only if $\mathsf{pte}(p)(k) = \mathfrak{t}$, with $p \in \mathrm{AP}$.
(2) For all formulas φ, φ_1, and φ_2, it holds that:
   (a) $\mathsf{pte}, k \models \neg\varphi$ if and only if not $\mathsf{pte}, k \models \varphi$, that is $\mathsf{pte}, k \not\models \varphi$;
   (b) $\mathsf{pte}, k \models \varphi_1 \wedge \varphi_2$ if and only if $\mathsf{pte}, k \models \varphi_1$ and $\mathsf{pte}, k \models \varphi_2$;
   (c) $\mathsf{pte}, k \models \varphi_1 \vee \varphi_2$ if and only if $\mathsf{pte}, k \models \varphi_1$ or $\mathsf{pte}, k \models \varphi_2$;
   (d) $\mathsf{pte}, k \models \mathsf{X}\,\varphi$ if and only if $\mathsf{pte}, k+1 \models \varphi$;
   (e) $\mathsf{pte}, k \models \mathsf{F}\,\varphi$ if and only if there is an index $i \in \mathbb{N}$ with $k \leq i$ such that $\mathsf{pte}, i \models \varphi$;
   (f) $\mathsf{pte}, k \models \mathsf{G}\,\varphi$ if and only if for all indexes $i \in \mathbb{N}$ with $k \leq i$, it holds that $\mathsf{pte}, i \models \varphi$.
(3) For an atomic proposition $q \in \mathrm{AP}$ and a formula φ, it holds that:
   (a) $\mathsf{pte}, k \models \exists q.\varphi$ if and only if there exists a temporal truth evaluation $\mathsf{tte} \in \mathrm{TTE}$ such that $\mathsf{pte}[q \mapsto \mathsf{tte}], k \models \varphi$;
   (b) $\mathsf{pte}, k \models \forall q.\varphi$ if and only if for all temporal truth evaluations $\mathsf{tte} \in \mathrm{TTE}$ it holds that $\mathsf{pte}[q \mapsto \mathsf{tte}], k \models \varphi$.

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 nonelementary hardness result for the satisfiability problem.

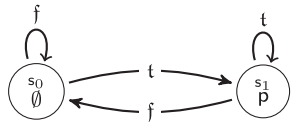


Fig. 6. The CGS $\mathcal{G}_{Rdc}$.

3.2. Nonelementary 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 constant size CGS with a unique atomic proposition.

To do this, we first prove the following auxiliary lemma, which actually represents the main step of the aforementioned reduction.

LEMMA 3.3 (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* SL[k-alt] *variable-closed formula* $\overline{\varphi}$ *such that* φ *is satisfiable if and only if* $\mathcal{G}_{Rdc}, \chi, s_0 \models \overline{\varphi}$ *for all complete assignments* $\chi \in \mathrm{Asg}(\mathrm{Ag}, s_0)$.

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 6, 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 $\tau(s, \mathsf{d}) = \mathsf{s}_0$ if and only if $\mathsf{d}(\alpha) = \mathsf{f}$ for all $s \in \mathrm{St}$ and $\mathsf{d} \in \mathrm{Dc}$. 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 if and only if $\overline{\varphi}$ 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 if and only if $\mathcal{G}_{Rdc}, \chi, \mathsf{s}_0 \models \overline{\varphi}$ 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 $\psi \in \mathsf{sub}(\varphi)$, propositional truth evaluations $\mathsf{pte} \in \mathrm{PTE}$, and $i \in \mathbb{N}$, it holds that $\mathsf{pte}, i \models \psi$ if and only if $\mathcal{G}_{Rdc}, (\chi, \mathsf{s}_0)^i \models \overline{\psi}$, for each total assignment $\chi \in \mathrm{Asg}(\{\alpha\} \cup \{\mathsf{x}_q \in \mathrm{Var} : q \in \mathsf{free}(\psi)\}, \mathsf{s}_0)$ such that $\chi(\mathsf{x}_q)((\pi)_{\leq n}) = \mathsf{pte}(q)(n)$, where $\pi \triangleq \mathsf{play}(\chi, \mathsf{s}_0)$ for all $q \in \mathsf{free}(\psi)$ 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 3.2 of QPTL semantics, we have that $\mathsf{pte}, i \models q$ if and only if $\mathsf{pte}(q)(i) = \mathsf{t}$. Thus, due to the aforementioned constraint on the assignment, it follows that $\mathsf{pte}, i \models q$ if and only if $\chi(\mathsf{x}_q)((\pi)_{\leq i}) = \mathsf{t}$. Now, by applying Items 4 and 5a of Definition 2.12 of SL semantics, we have that $\mathcal{G}_{Rdc}, (\chi, \mathsf{s}_0)^i \models (\alpha, \mathsf{x}_q)\mathsf{X}\,\mathsf{p}$ if and only if $\mathcal{G}_{Rdc}, (\chi'[\alpha \mapsto \chi'(\mathsf{x}_q)], s')^1 \models \mathsf{p}$, 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 $\mathcal{G}_{Rdc}, (\chi'[\alpha \mapsto \chi'(\mathsf{x}_q)], s')^1 \models \mathsf{p}$ if and only if $(\pi')_1 = \mathsf{s}_1$, 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, $\mathcal{G}_{Rdc}, (\chi, \mathsf{s}_0)^i \models (\alpha, \mathsf{x}_q)\mathsf{X}\,\mathsf{p}$ if and only if $\chi'(\mathsf{x}_q)((\pi')_{\leq 0}) = \mathsf{t}$. Now, by observing that $(\pi')_{\leq 0} = (\pi)_i$ and using the aforementioned definition of χ', we obtain that $\chi'(\mathsf{x}_q)((\pi')_{\leq 0}) = \chi(\mathsf{x}_q)((\pi)_{\leq i})$. Hence, $\mathsf{pte}, i \models q$ if and only if $\mathsf{pte}(q)(i) = \chi(\mathsf{x}_q)((\pi)_{\leq i}) = \mathsf{t} = \chi'(\mathsf{x}_q)((\pi')_{\leq 0})$ if and only if $\mathcal{G}_{Rdc}, (\chi, \mathsf{s}_0)^i \models (\alpha, \mathsf{x}_q)\mathsf{X}\,\mathsf{p}$.

—$\psi = \exists q.\psi'$.

[Only if]. If $\mathsf{pte}, i \models \exists q.\psi'$, by Item 3a of Definition 3.2, 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 3.2, 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 3.2, 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 3.2, 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 nonelementary lower bound for the model-checking problem of SL.

THEOREM 3.4 (SL MODEL-CHECKING HARDNESS). *The model-checking problem for* SL[*k*-alt] *is* *k*-EXPSPACE-HARD.

PROOF. Let φ be a QPTL[*k*-alt] sentence, $\overline{\varphi}$ the related SL[*k*-alt] variable-closed formula, and $\mathcal{G}_{Rdc}$ the CGS of Lemma 3.3 of QPTL reduction. Then, by applying the previously mentioned lemma, it is easy to see that φ is satisfiable if and only if $\mathcal{G}_{Rdc} \models [[\mathsf{x}]](\alpha, \mathsf{x})\overline{\varphi}$ if and only if $\mathcal{G}_{Rdc} \models \langle\langle \mathsf{x} \rangle\rangle(\alpha, \mathsf{x})\overline{\varphi}$. Thus, the satisfiability problem for QPTL can be reduced to the model-checking problem for SL. Now, because the satisfiability problem for QPTL[*k*-alt] is *k*-EXPSPACE-HARD [Sistla et al. 1987], we have that the model-checking problem for SL[*k*-alt] is *k*-EXPSPACE-HARD as well. □

4. STRATEGY QUANTIFICATIONS

Because model checking for SL is nonelementary hard, while the same problem for ATL* is only 2EXPTIME-COMPLETE, a question that arises naturally is whether there are proper fragments of SL of practical interest that strictly subsuming ATL*yet still have the same elementary complexity. In this section, we answer positively to this question

and go even further. More precisely, we reveal a fundamental property that, if satisfied, allows to retain a 2ExpTime-complete model-checking problem. This is the property of *behavioral* quantification. Standard quantification over strategies introduces a somewhat unintuitive interpretation in the semantics of games because it instantiates full strategies, which also includes future actions, whereas players see only actions played in the past. In more detail, quantification of the form $\forall\exists$ means that the existentially quantified strategy is chosen in response to the universally quantified strategy, which prescribes actions in response to histories that may never arise in the actual game. On the other hand, in classic games, the player reacts only to current and past actions of other players, so the response of the existentially quantified strategy to a given history should depend only on the response of the universally quantified strategy to this history. To formally introduce the concept of behavioral quantification, we use the notion of the *Skolem dependence function* as a machinery (see also Kolaitis [1985] for more on this subject).

The remaining part of this section is organized as follows. In Section 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 Section 4.2, we define the notion of the Skolem dependence function, which is used in Section 4.3 to introduce the concept of behavioral. Finally, in Section 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

To formalize the syntactic fragments of Sl we want to investigate, we first need to define the concepts of *quantification* and *binding prefixes*. A *quantification prefix* over a set $\mathrm{V} \subseteq \mathrm{Var}$ of variables is a finite word $\wp \in \{\langle\!\langle x \rangle\!\rangle, [\![x]\!] : x \in \mathrm{V}\}^{|\mathrm{V}|}$ of length $|\mathrm{V}|$ such that each variable $x \in \mathrm{V}$ occurs just once in $\wp$, that is, there is exactly one index $i \in [0, |\mathrm{V}|[$ such that $(\wp)_i \in \{\langle\!\langle x \rangle\!\rangle, [\![x]\!]\}$.

A *binding prefix* over a set of variables $\mathrm{V} \subseteq \mathrm{Var}$ is a finite word $\flat \in \{(a, x) : a \in \mathrm{Ag} \wedge x \in \mathrm{V}\}^{|\mathrm{Ag}|}$ of length $|\mathrm{Ag}|$ such that each agent $a \in \mathrm{Ag}$ occurs just once in $\flat$, that is, there is exactly one index $i \in [0, |\mathrm{Ag}|[$ for which $(\flat)_i \in \{(a, x) : x \in \mathrm{V}\}$.

Finally, $\mathrm{Qnt}(\mathrm{V}) \subseteq \{\langle\!\langle x \rangle\!\rangle, [\![x]\!] : x \in \mathrm{V}\}^{|\mathrm{V}|}$ and $\mathrm{Bnd}(\mathrm{V}) \subseteq \{(a, x) : a \in \mathrm{Ag} \wedge x \in \mathrm{V}\}^{|\mathrm{Ag}|}$ denote, respectively, the sets of all quantification and binding prefixes over variables in V.

We now have all the tools to define the syntactic fragments we want to analyze, which we name *Nested-Goal*, *Boolean-Goal*, and *One-Goal Strategy Logic*, respectively (Sl[ng], Sl[bg], and Sl[1g]). For a *goal,* we mean an Sl agent-closed formula of the kind $\flat\varphi$, with $\mathrm{Ag} \subseteq \mathsf{free}(\varphi)$, being $\flat \in \mathrm{Bnd}(\mathrm{Var})$ 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 we 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 4.1 (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 $\mathrm{V} \subseteq \mathrm{Var}$, and binding prefixes Bnd(Var) by using the following grammar, with $p \in \mathrm{AP}$, $\wp \in \cup_{\mathrm{V} \subseteq \mathrm{Var}} \mathrm{Qnt}(\mathrm{V})$, and $\flat \in \mathrm{Bnd}(\mathrm{Var})$:

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

where in the formation rule $\wp\varphi$ it is ensured that φ is agent-closed and $\wp \in \mathrm{Qnt}(\mathsf{free}(\varphi))$.

In addition, SL[BG] formulas are determined by splitting the aforementioned 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 $\wp \in \mathsf{Qnt}(\mathsf{free}(\psi))$.

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 $\wp \in \mathsf{Qnt}(\mathsf{free}(\flat\varphi))$.

SL $\supset$ SL[NG] $\supset$ SL[BG] $\supset$ 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 SL[NG], SL[BG], and 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 $\wp'\psi' \in \mathsf{snt}(\wp\psi)$.

We want to remark that the nesting of goals in SL formulas is much more expressive than the nesting of quantification in ATL*. Indeed, in ATL*, we can make only a nesting of quantifications that are in some sense complete with respect to players, which implies, at every nesting, a total replacement of the strategy profile for all players. This is because of the fact that the quantifier operator over a set of agents A in ATL* includes implicitly the quantification of strategies for agents not in A, that is, each quantifier ranges over the whole set of agents, and the binding is immediately applied to the agents. Instead, in the case of SL, the quantification is referred to a single strategy and the binding with an agent can be postponed. This power of nesting allows us to embed very expressive logics such as QPTL, as shown in Section 3.

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

We now introduce two other general restrictions in which the numbers |Ag| of agents and |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 SL[n-ag], SL[m-var], and 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., SL[1-ag, fvs] = SL[1-ag]).

To start to practice with the aforementioned fragments, consider again the sentence φ of Example 2.13. It is easy to see that it actually belongs to SL[BG, 2-ag, 3-var, 2-alt] and to SL[NG], but not to SL[1G] because 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 $\flat_2 = (\alpha, \mathsf{y})(\beta, \mathsf{z})$.

In this article, we sometimes assert that a given formula φ belongs to an SL syntactic fragment if its syntax does not precisely correspond to what is described by the relative grammar. We do this 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 2.7 representing the existence of a Nash equilibrium. This formula is considered to belong to SL[BG] because it can be easily translated in the form $\phi_{NE} = \wp \bigwedge_{i=1}^{n} \flat_i \psi_i \rightarrow \flat \psi_i$, 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 $\mathsf{free}(\psi_i) =$ Ag. As another example, consider the sentence φ_{SP} of Example 2.8 representing the existence of a stability profile. This formula is also considered to belong to SL[BG] because it is equivalent to $\phi_{SP} = \wp \bigwedge_{i,j=1, i \neq j}^{n} \flat \psi_j \rightarrow ((\flat \psi_i \leftrightarrow \flat_i \psi_i) \rightarrow \flat_i \psi_j)$. Furthermore, in Example 2.15, it is immediate to see that the formulas φ_1 and φ_2 are in SL[1G, 4-ag, 4-var, fvs, 1-alt], while φ_3 is in SL[1G, 4-ag, 4-var, fvs, 2-alt] 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 SL[1G, fvs, 0-alt] and SL[1G, fvs, 1-alt], respectively. Moreover, GL [Alur et al. 2002] is the very simple fragment of SL[BG, fvs, 1-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 CHP-SL is the SL[BG, 2-ag, fvs] fragment interpreted over turn-based CGS.

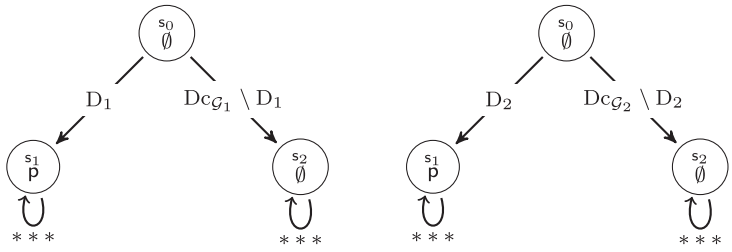
In the remainder of the article, we refer to CTL*, ATL*, and CHP-SL formulas by denoting them with their translation in SL.

Remark 4.2 (SL[NG] *Model-Checking Hardness*). It is well known that the nonelementary 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 3.3 of QPTL reduction puts QPTL[k-alt] sentences φ in prenex normal form into SL[NG, 1-ag, k-alt] variable-closed formulas $\overline{\varphi} = \wp \psi$. Moreover, the derived SL[1-ag, k-alt] sentence $\langle\langle \mathsf{x} \rangle\rangle (\alpha, \mathsf{x}) \wp \psi$ used in Theorem 3.4 of SL model-checking hardness is equivalent to the SL[NG, 1-ag, k-alt] principal sentence $\langle\langle \mathsf{x} \rangle\rangle \wp (\alpha, \mathsf{x}) \psi$ because x is not used in the quantification prefix $\wp$. Thus, the hardness result for the model-checking problem holds for SL[NG, 1-ag, k-alt] as well. However, it is important to observe that, unfortunately, it is not know if such an hardness result holds for SL[BG] and, in particular, for CHP-SL. We leave this problem open here.

Differently from SL[1G], in fragments like SL[BG], we are able to associate different strategies with the same agent. This feature allows to represent game properties such as Nash equilibria, equilibrium profiles, and the like. This syntactic flexibility, however, is not the feature that forces these fragments to have nonelementary complexities for the model-checking problem. Indeed, as it has been recently pointed out in Mogavero et al. [2013], the same problem for some of the SL fragments preserving this feature is 2EXPTIME-COMPLETE. Thus, the nonelementary hardness in the model-checking problem for SL[NG] is due to the ability to nest the binding of strategies by means of temporal operators. A deeper study of these aspects is out of the scope of this article.

At this point, we prove that ATL* is strictly less expressive than SL[1G] and, consequently, than SL[BG] and SL[NG]. To do this, we show the existence of two structures that result to be equivalent only with respect to sentences having an 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 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 SL, which would allow to determine in a more direct way whether two structures are equivalent with respect to a particular SL fragment.

THEOREM 4.3 (SL[1G] VERSUS ATL* EXPRESSIVENESS). *There exists an* SL[1G, 3-ag, fvs, 2-alt] *sentence having no* ATL* *equivalent sentence.*



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

Fig. 7. Alternation-2 nonequivalent CGSS.

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 7, 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 SL[1G, 3-ag, fvs, 2-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, because of 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 $j \in [1, \omega[$, that is, 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 ATL* formula $\wp\flat\psi$, there is a literal $l_\psi \in \{\mathsf{p}, \neg\mathsf{p}\}$ such that $\mathcal{G}_i \models \wp\flat\psi$ if and only if $\mathcal{G}_i \models \wp\flat\mathsf{X}l_\psi$ for all $i \in \{1, 2\}$. Without loss of generality, we can suppose that $\flat = \flat^*$ because 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 $\wp_1 \triangleq [[\mathsf{x}]][[z]]\langle\langle\mathsf{y}\rangle\rangle$, $\wp_2 \triangleq \langle\langle\mathsf{x}\rangle\rangle\langle\langle y\rangle\rangle[[\mathsf{z}]]$, and $\wp_3 \triangleq [[\mathsf{y}]][[z]]\langle\langle\mathsf{x}\rangle\rangle$. Thus, to prove that every 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. Skolem Dependence Functions

We now introduce the concept of Skolem dependence function (Sdf) of a quantification and show how any quantification prefix contained into an SL formula can be represented by an adequate choice of a Skolem dependence function over strategies. The main idea here is inspired by what Skolem proposed for the first-order logic to eliminate all existential quantifications over variables, by substituting them with second-order existential quantifications over functions, whose choice is uniform with respect to the universal variables.

First, we introduce some notation regarding quantification prefixes. Let $\wp \in \mathrm{Qnt(V)}$ 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 $[[\wp]] \triangleq \mathrm{V}(\wp)\backslash\langle\langle\wp\rangle\rangle$, we denote the sets of existential and universal variables quantified in $\wp$, respectively. 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 $\wp$, that is, 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 $x <_{\wp} y$, that is, 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, that is, for all indexes $i \in [0, |\wp|[$, it holds that $(\overline{\wp})_i = \langle\!\langle x \rangle\!\rangle$ if and only if $(\wp)_i = [\![x]\!]$, 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 $\wp = [\![\mathsf{x}]\!]\langle\!\langle \mathsf{y} \rangle\!\rangle\langle\!\langle \mathsf{z} \rangle\!\rangle[\![\mathsf{w}]\!]\langle\!\langle \mathsf{v} \rangle\!\rangle$. Then, we have $\langle\!\langle \wp \rangle\!\rangle = \{\mathsf{y}, \mathsf{z}, \mathsf{v}\}$, $[\![\wp]\!] = \{\mathsf{x}, \mathsf{w}\}$, $\mathrm{Dep}(\wp, \mathsf{x}) = \mathrm{Dep}(\wp, \mathsf{w}) = \emptyset$, $\mathrm{Dep}(\wp, \mathsf{y}) = \mathrm{Dep}(\wp, \mathsf{z}) = \{\mathsf{x}\}$, $\mathrm{Dep}(\wp, \mathsf{v}) = \{\mathsf{x}, \mathsf{w}\}$, and $\overline{\wp} = \langle\!\langle \mathsf{x} \rangle\!\rangle[\![\mathsf{y}]\!][\![\mathsf{z}]\!]\langle\!\langle \mathsf{w} \rangle\!\rangle[\![\mathsf{v}]\!]$.

Finally, we define the notion of *valuation* of variables over a generic set D, called *domain*, that is, a partial function $\mathsf{v} : \mathrm{Var} \rightharpoonup \mathrm{D}$ mapping every variable in its domain to an element in D. By $\mathrm{Val}_{\mathrm{D}}(\mathrm{V}) \triangleq \mathrm{V} \to \mathrm{D}$, 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 the *Skolem dependence function*.

Definition 4.4 (*Skolem Dependence Function*). 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 *Skolem dependence function* for $\wp$ over D is a function $\theta : \mathrm{Val}_{\mathrm{D}}([\![\wp]\!]) \to \mathrm{Val}_{\mathrm{D}}(\mathrm{V})$ satisfying the following properties:

(1) $\theta(\mathsf{v})_{\restriction [\![\wp]\!]} = \mathsf{v}$ for all $\mathsf{v} \in \mathrm{Val}_{\mathrm{D}}([\![\wp]\!])$;[11]
(2) $\theta(\mathsf{v}_1)(x) = \theta(\mathsf{v}_2)(x)$ for all $\mathsf{v}_1, \mathsf{v}_2 \in \mathrm{Val}_{\mathrm{D}}([\![\wp]\!])$ and $x \in \langle\!\langle \wp \rangle\!\rangle$ such that $\mathsf{v}_{1 \restriction \mathrm{Dep}(\wp, x)} = \mathsf{v}_{2 \restriction \mathrm{Dep}(\wp, x)}$.

$\mathrm{SDF}_{\mathrm{D}}(\wp)$ denotes the set of all Skolem dependence functions for $\wp$ over D.

Intuitively, Item 1 asserts that θ takes the same values of its argument with respect to the universal variables in $\wp$, and Item 2 ensures that the value of θ with respect to 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 Skolem dependence function θ for $\wp$ can be considered as a set of classical *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 with respect to the order of quantifications. Observe that each $\theta \in \mathrm{SDF}_{\mathrm{D}}(\wp)$ is injective, so $|\mathsf{rng}(\theta)| = |\mathsf{dom}(\theta)| = |\mathrm{D}|^{|[\![\wp]\!]|}$. Moreover, $|\mathrm{SDF}_{\mathrm{D}}(\wp)| = \prod_{x \in \langle\!\langle \wp \rangle\!\rangle} |\mathrm{D}|^{|\mathrm{D}|^{|\mathrm{Dep}(\wp, x)|}}$. As an example, let $\mathrm{D} = \{0, 1\}$ and $\wp = [\![\mathsf{x}]\!]\langle\!\langle \mathsf{y} \rangle\!\rangle[\![\mathsf{z}]\!] \in \mathrm{Qnt}(\mathrm{V})$ be a quantification prefix over $\mathrm{V} = \{\mathsf{x}, \mathsf{y}, \mathsf{z}\}$. Then, we have that $|\mathrm{SDF}_{\mathrm{D}}(\wp)| = 4$ and $|\mathrm{SDF}_{\mathrm{D}}(\overline{\wp})| = 8$. Moreover, the Skolem dependence functions $\theta_i \in \mathrm{SDF}_{\mathrm{D}}(\wp)$ with $i \in [0, 3]$ and $\overline{\theta}_i \in \mathrm{SDF}_{\mathrm{D}}(\overline{\wp})$ with $i \in [0, 7]$, for a particular fixed order, are such that $\theta_0(\mathsf{v})(\mathsf{y}) = 0$, $\theta_1(\mathsf{v})(\mathsf{y}) = \mathsf{v}(\mathsf{x})$, $\theta_2(\mathsf{v})(\mathsf{y}) = 1 - \mathsf{v}(\mathsf{x})$, and $\theta_3(\mathsf{v})(\mathsf{y}) = 1$ for all $\mathsf{v} \in \mathrm{Val}_{\mathrm{D}}([\![\wp]\!])$, and $\overline{\theta}_i(\overline{\mathsf{v}})(\mathsf{x}) = 0$ with $i \in [0, 3]$, $\overline{\theta}_i(\overline{\mathsf{v}})(\mathsf{x}) = 1$ with $i \in [4, 7]$, $\overline{\theta}_0(\overline{\mathsf{v}})(\mathsf{z}) = \overline{\theta}_4(\overline{\mathsf{v}})(\mathsf{z}) = 0$, $\overline{\theta}_1(\overline{\mathsf{v}})(\mathsf{z}) = \overline{\theta}_5(\overline{\mathsf{v}})(\mathsf{z}) = \overline{\mathsf{v}}(\mathsf{y})$, $\overline{\theta}_2(\overline{\mathsf{v}})(\mathsf{z}) = \overline{\theta}_6(\overline{\mathsf{v}})(\mathsf{z}) = 1 - \overline{\mathsf{v}}(\mathsf{y})$, and $\overline{\theta}_3(\overline{\mathsf{v}})(\mathsf{z}) = \overline{\theta}_7(\overline{\mathsf{v}})(\mathsf{z}) = 1$ for all $\overline{\mathsf{v}} \in \mathrm{Val}_{\mathrm{D}}([\![\overline{\wp}]\!])$.

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 Skolem dependence function over strategies. This procedure can be seen as the equivalent of *Skolemization* in first-order logic (see Hodges [1993] for more details).

THEOREM 4.5 (SL STRATEGY QUANTIFICATION). *Let $\mathcal{G}$ be a* CGS *and $\varphi = \wp\psi$ an* SL *formula, being $\wp \in \mathrm{Qnt}(\mathrm{V})$ a quantification prefix over a set $\mathrm{V} \subseteq \mathsf{free}(\psi) \cap \mathrm{Var}$ of variables. Then, for all assignments $\chi \in \mathrm{Asg}(\mathsf{free}(\varphi), s_0)$, the following holds: $\mathcal{G}, \chi, s_0 \models \varphi$ if and only if*

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

there exists a Skolem dependence function $\theta \in \mathrm{SDF}_{\mathrm{Str}(s_0)}(\wp)$ *such that* $\mathcal{G}, \chi \Cup \theta(\chi'), s_0 \models \psi$ *for all* $\chi' \in \mathrm{Asg}([\![\wp]\!], s_0)$.[12]

Proof. The proof proceeds by induction on the length of the quantification prefix $\wp$. For the base case $|\wp| = 0$, the thesis immediately follows because $[\![\wp]\!] = \emptyset$ and, consequently, both $\mathrm{SDF}_{\mathrm{Str}(s_0)}(\wp)$ 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 $\wp = \mathsf{Qn} \cdot \wp'$. Then, two possible cases arise: either $\mathsf{Qn} = \langle\!\langle x \rangle\!\rangle$ or $\mathsf{Qn} = [\![x]\!]$.

—$\mathsf{Qn} = \langle\!\langle x \rangle\!\rangle$.
By Item 3a of Definition 2.12 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 Skolem dependence function $\theta \in \mathrm{SDF}_{\mathrm{Str}(s_0)}(\wp')$ such that $\mathcal{G}, \chi[x \mapsto \mathsf{f}] \Cup \theta(\chi'), s_0 \models \psi$ for all $\chi' \in \mathrm{Asg}([\![\wp']\!], s_0)$. Now, consider the function $\theta' : \mathrm{Asg}([\![\wp]\!], s_0) \to \mathrm{Asg}(\mathrm{V}, s_0)$ 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 Skolem dependence function for $\wp$ over $\mathrm{Str}(s_0)$, i.e., $\theta' \in \mathrm{SDF}_{\mathrm{Str}(s_0)}(\wp)$. Moreover, $\chi[x \mapsto \mathsf{f}] \Cup \theta(\chi') = \chi \Cup \theta(\chi')[x \mapsto \mathsf{f}] = \chi \Cup \theta'(\chi')$, for $\chi' \in \mathrm{Asg}([\![\wp]\!], s_0)$. Hence, $\mathcal{G}, \chi \Cup \theta'(\chi'), s_0 \models \psi$ for all $\chi' \in \mathrm{Asg}([\![\wp]\!], s_0)$.

—$\mathsf{Qn} = [\![x]\!]$.
By Item 3b of Definition 2.12, 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 Skolem dependence function $\theta_{\mathsf{f}} \in \mathrm{SDF}_{\mathrm{Str}(s_0)}(\wp')$ such that $\mathcal{G}, \chi[x \mapsto \mathsf{f}] \Cup \theta_{\mathsf{f}}(\chi'), s_0 \models \psi$ for all $\chi' \in \mathrm{Asg}([\![\wp']\!], s_0)$. Now, consider the function $\theta' : \mathrm{Asg}([\![\wp]\!], s_0) \to \mathrm{Asg}(\mathrm{V}, s_0)$ defined by $\theta'(\chi') \triangleq \theta_{\chi'(x)}(\chi'_{\restriction [\![\wp']\!]})[x \mapsto \chi'(x)]$ for all $\chi' \in \mathrm{Asg}([\![\wp]\!], s_0)$. It is evident that θ' is a Skolem dependence function for $\wp$ over $\mathrm{Str}(s_0)$, that is, $\theta' \in \mathrm{SDF}_{\mathrm{Str}(s_0)}(\wp)$. Moreover, $\chi[x \mapsto \mathsf{f}] \Cup \theta_{\mathsf{f}}(\chi') = \chi \Cup \theta_{\mathsf{f}}(\chi')[x \mapsto \mathsf{f}] = \chi \Cup \theta'(\chi'[x \mapsto \mathsf{f}])$ for $\mathsf{f} \in \mathrm{Str}(s_0)$ and $\chi' \in \mathrm{Asg}([\![\wp']\!], s_0)$. Hence, $\mathcal{G}, \chi \Cup \theta'(\chi'), s_0 \models \psi$ for all $\chi' \in \mathrm{Asg}([\![\wp]\!], s_0)$.

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

—$\mathsf{Qn} = \langle\!\langle x \rangle\!\rangle$.
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 $\theta' : \mathrm{Asg}([\![\wp']\!], s_0) \to \mathrm{Asg}(\mathrm{V} \setminus \{x\}, s_0)$ defined by $\theta'(\chi') \triangleq \theta(\chi')_{\restriction (\mathrm{V} \setminus \{x\})}$ for all $\chi' \in \mathrm{Asg}([\![\wp']\!], s_0)$. It is easy to check that θ' is a Skolem dependence function for $\wp'$ over $\mathrm{Str}(s_0)$, that is, $\theta' \in \mathrm{SDF}_{\mathrm{Str}(s_0)}(\wp')$. Moreover, $\chi \Cup \theta(\chi') = \chi \Cup \theta'(\chi')[x \mapsto \mathsf{f}] = \chi[x \mapsto \mathsf{f}] \Cup \theta'(\chi')$ for $\chi' \in \mathrm{Asg}([\![\wp']\!], s_0)$. Then, it is evident that $\mathcal{G}, \chi[x \to \mathsf{f}] \Cup \theta'(\chi'), s_0 \models \psi$ 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, by Item 3a of Definition 2.12, means that $\mathcal{G}, \chi, s_0 \models \varphi$.

—$\mathsf{Qn} = [\![x]\!]$.
First note that $[\![\wp]\!] = [\![\wp']\!] \cup \{x\}$. Also, consider the functions $\theta'_{\mathsf{f}} : \mathrm{Asg}([\![\wp']\!], s_0) \to \mathrm{Asg}(\mathrm{V}\setminus\{x\}, s_0)$ defined by $\theta'_{\mathsf{f}}(\chi') \triangleq \theta(\chi'[x \mapsto \mathsf{f}])_{\restriction (\mathrm{V}\setminus\{x\})}$ for each $\mathsf{f} \in \mathrm{Str}(s_0)$ and $\chi' \in \mathrm{Asg}([\![\wp']\!], s_0)$. It is easy to see that every θ'_{f} is a Skolem dependence function for $\wp'$ over $\mathrm{Str}(s_0)$, that is, $\theta'_{\mathsf{f}} \in \mathrm{SDF}_{\mathrm{Str}(s_0)}(\wp')$. Moreover, $\chi \Cup \theta(\chi') = \chi \Cup \theta'_{\chi'(x)}(\chi'_{\restriction [\![\wp']\!]})[x \mapsto$

[12]By $\mathsf{g}_1 \Cup \mathsf{g}_2 : (\mathrm{X}_1 \cup \mathrm{X}_2) \to (\mathrm{Y}_1 \cup \mathrm{Y}_2)$, we denote the operation of union of two functions $\mathsf{g}_1 : \mathrm{X}_1 \to \mathrm{Y}_1$ and $\mathsf{g}_2 : \mathrm{X}_2 \to \mathrm{Y}_2$ defined on disjoint domains (i.e., $\mathrm{X}_1 \cap \mathrm{X}_2 = \emptyset$).

$\chi'(x)] = \chi[x \mapsto \chi'(x)] \uplus \theta'_{\chi'(x)}(\chi'_{\restriction [\![\wp']\!]})$ for $\chi' \in \mathrm{Asg}([\![\wp]\!], s_0)$. Then, it is evident that $\mathcal{G}, \chi[x \to \mathsf{f}] \uplus \theta'_{\mathsf{f}}(\chi'), s_0 \models \psi$ for all $\mathsf{f} \in \mathrm{Str}(s_0)$ and $\chi' \in \mathrm{Asg}([\![\wp']\!], s_0)$. By the inductive hypothesis, we derive that $\mathcal{G}, \chi[x \mapsto \mathsf{f}], s_0 \models \wp'\psi$ for all $\mathsf{f} \in \mathrm{Str}(s_0)$, which, by Item 3b of Definition 2.12, means that $\mathcal{G}, \chi, s_0 \models \varphi$.

Thus, the thesis of the theorem holds. □

As an immediate consequence of the previous result, we derive the following corollary that restricts to SL sentences.

COROLLARY 4.6 (SL STRATEGY QUANTIFICATION). *Let $\mathcal{G}$ be a* CGS *and $\varphi = \wp\psi$ an* SL *sentence, where ψ is agent-closed and $\wp \in \mathrm{Qnt}(\mathsf{free}(\psi))$. Then, $\mathcal{G} \models \varphi$ if and only if there exists a Skolem dependence function $\theta \in \mathrm{SDF}_{\mathrm{Str}(s_0)}(\wp)$ such that $\mathcal{G}, \theta(\chi), s_0 \models \psi$ for all $\chi \in \mathrm{Asg}([\![\wp]\!], s_0)$.*

4.3. Behavioral Quantifications

We now have all tools we need to introduce the property of behavioral for a particular class of Skolem dependence functions. Intuitively, a Skolem dependence function over functions from a set T to a set D is behavioral if it can be split into a set of Skolem dependence functions over D, one for each element of T. This idea allows us to enormously simplify the reasoning about strategy quantifications because 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 Skolem dependence function $\theta \in \mathrm{SDF}_{\mathrm{Str}(s)}(\wp)$ over strategies in a function $\widetilde{\theta} : \mathrm{Trk}(s) \to \mathrm{SDF}_{\mathrm{Ac}}(\wp)$ that associates with each track a Skolem dependence function over actions.

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

Definition 4.7 (*Adjoint Functions*). Let D, T, U, and V be four sets, and $\mathsf{m} : (\mathrm{T} \to \mathrm{D})^{\mathrm{U}} \to (\mathrm{T} \to \mathrm{D})^{\mathrm{V}}$ and $\widetilde{\mathsf{m}} : \mathrm{T} \to (\mathrm{D}^{\mathrm{U}} \to \mathrm{D}^{\mathrm{V}})$ two functions. Then, $\widetilde{\mathsf{m}}$ is the adjoint of m if $\widetilde{\mathsf{m}}(t)(\widehat{\mathsf{g}}(t))(x) = \mathsf{m}(\mathsf{g})(x)(t)$ for all $\mathsf{g} \in (\mathrm{T} \to \mathrm{D})^{\mathrm{U}}$, $x \in \mathrm{V}$, and $t \in \mathrm{T}$[13]

Intuitively, $\widetilde{\mathsf{m}}$ 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 $\mathsf{g}_1, \mathsf{g}_2 \in (\mathrm{T} \to \mathrm{D})^{\mathrm{U}}$ such that $\widehat{\mathsf{g}_1}(t) = \widehat{\mathsf{g}_2}(t)$ for some $t \in \mathrm{T}$, it holds that $\mathsf{m}(\mathsf{g}_1)(x)(t) = \mathsf{m}(\mathsf{g}_2)(x)(t)$ for all $x \in \mathrm{V}$. It is immediate to observe that if a function has an adjoint then this adjoint is unique. In 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.

The next lemma formally states the property briefly described earlier, that is, that each Skolem dependence function over a set $\mathrm{T} \to \mathrm{D}$, admitting an adjoint function, can be represented as a function, with T as domain, which returns Skolem dependence functions over D as values.

LEMMA 4.8 (ADJOINT SKOLEM DEPENDENCE FUNCTIONS). *Let $\wp \in \mathrm{Qnt}(\mathrm{V})$ be a quantification prefix over a set $\mathrm{V} \subseteq \mathrm{Var}$ of variables,* D *and* T *two sets, and $\theta : \mathrm{Val}_{\mathrm{T}\to\mathrm{D}}([\![\wp]\!]) \to \mathrm{Val}_{\mathrm{T}\to\mathrm{D}}(\mathrm{V})$ and $\widetilde{\theta} : \mathrm{T} \to (\mathrm{Val}_{\mathrm{D}}([\![\wp]\!]) \to \mathrm{Val}_{\mathrm{D}}(\mathrm{V}))$ two functions such that $\widetilde{\theta}$ is the adjoint of θ. Then, $\theta \in \mathrm{SDF}_{\mathrm{T}\to\mathrm{D}}(\wp)$ if and only if for all $t \in \mathrm{T}$, it holds that $\widetilde{\theta}(t) \in \mathrm{SDF}_{\mathrm{D}}(\wp)$.*

We now define the formal meaning of the behavioral of a Skolem dependence function over functions.

[13]By $\widehat{\mathsf{g}} : \mathrm{Y} \to \mathrm{X} \to \mathrm{Z}$, we denote the operation of flipping of a function $\mathsf{g} : \mathrm{X} \to \mathrm{Y} \to \mathrm{Z}$.

Definition 4.9 (*Behavioral Skolem Dependence Functions*). Let $\wp \in \mathrm{Qnt}(\mathrm{V})$ be a quantification prefix over a set $\mathrm{V} \subseteq \mathrm{Var}$ of variables, D and T two sets, and $\theta \in \mathrm{SDF}_{\mathrm{T}\to\mathrm{D}}(\wp)$ a Skolem dependence function for $\wp$ over $\mathrm{T} \to \mathrm{D}$. Then, θ is behavioral if it admits an adjoint function. $\mathrm{BSDF}_{\mathrm{T}\to\mathrm{D}}(\wp)$ denotes the set of all behavioral Skolem dependence functions for $\wp$ over $\mathrm{T} \to \mathrm{D}$.

It is important to observe that, unfortunately, there are Skolem dependence functions that are not behavioral. To easily understand why this is actually the case, it is enough to count both the number of Skolem dependence functions $\mathrm{SDF}_{\mathrm{T}\to\mathrm{D}}(\wp)$ and those of adjoint functions $\mathrm{T} \to \mathrm{SDF}_{\mathrm{D}}(\wp)$, 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 $|\mathrm{SDF}_{\mathrm{T}\to\mathrm{D}}(\wp)| = \prod_{x\in\langle\!\langle\wp\rangle\!\rangle} |\mathrm{D}|^{|\mathrm{T}|\cdot|\mathrm{D}|^{|\mathrm{T}|\cdot|\mathrm{Dep}(\wp,x)|}} > \prod_{x\in\langle\!\langle\wp\rangle\!\rangle} |\mathrm{D}|^{|\mathrm{T}|\cdot|\mathrm{D}|^{|\mathrm{Dep}(\wp,x)|}} = |\mathrm{T} \to \mathrm{SDF}_{\mathrm{D}}(\wp)|$. So, there are much more Skolem dependence functions, 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 behavioral Skolem dependence functions over $\mathrm{T} \to \mathrm{D}$, that is, it is such that each $\theta \in \mathrm{SDF}_{\mathrm{T}\to\mathrm{D}}(\wp)$ is behavioral for all $\wp \in \mathrm{Qnt}_{\exists^*\forall^*}(\mathrm{V})$. This is because of the fact that there are no functional dependencies between variables, that is, for each $x \in \langle\!\langle \wp \rangle\!\rangle$, it holds that $\mathrm{Dep}(\wp, x) = \emptyset$.

Finally, we introduce an important semantics for syntactic fragments of SL, which is based on the concept of a behavioral Skolem dependence function over strategies, and we refer to the related satisfiability concept as *behavioral satisfiability*, in symbols $\models_{\mathrm{B}}$. Intuitively, such a semantics has the peculiarity that a strategy used in an existential quantification, to satisfy a formula, is only chosen between those that are behavioral with respect to 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 for the classic semantics, instead. This means that c does not depend on the values of the other strategies on tracks ρ' that extend ρ, that is, it does not depend on future choices made on ρ'. In addition, we have that c does not depend on values of parallel tracks ρ' that only share a prefix with ρ, that is, it is independent on choices made on the possibly alternative futures ρ'. The behavioral 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 $\models_{\mathrm{B}}$, and we omit to report it here. In the following definition, we only describe the part concerning the quantification prefixes.

Definition 4.10 (SL[NG] *Behavioral 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 $\wp \in \mathrm{Qnt}(\mathsf{free}(\psi))$. Then, $\mathcal{G}, \varnothing, s \models_{\mathrm{B}} \wp\psi$ if there is a behavioral Skolem dependence function $\theta \in \mathrm{BSDF}_{\mathrm{Str}(s)}(\wp)$ for $\wp$ over $\mathrm{Str}(s)$ such that $\mathcal{G}, \theta(\chi), s \models_{\mathrm{B}} \psi$ for all $\chi \in \mathrm{Asg}([\![\wp]\!], s)$.

It is immediate to see a strong similarity between the statement of Corollary 4.6 of the SL strategy quantification and the previous definition. The only crucial difference resides in the choice of the kind of Skolem dependence function. Moreover, observe that, differently from the classic semantics, the quantifications in the prefix are not treated individually but as an atomic block. This is because of the necessity of having a strict correlation between the point-wise structure of the quantified strategies.

Remark 4.11 (SL *Behavioral Semantics*). It can be interesting to know that we do not define a behavioral semantics for the whole SL because we are currently not able to easily use the concept of behavioral Skolem dependence function when the quantifications are not necessarily grouped in prefixes, that is, when the formula is

not in prenex normal form. In fact, this may represent a challenging problem, whose solution is left to future work.

Because of the new semantics of SL[NG], we have to redefine the related concepts of model and satisfiability to differentiate between the classic relation $\models$ and the behavioral one $\models_{\mathrm{B}}$. Indeed, as we show later, there are sentences that are satisfiable but not behavioral satisfiable. We say that a CGS $\mathcal{G}$ is a behavioral model of an SL[NG] sentence φ, in symbols $\mathcal{G} \models_{\mathrm{B}} \varphi$, if $\mathcal{G}, \varnothing, s_0 \models_{\mathrm{B}} \varphi$. In general, we also say that $\mathcal{G}$ is a behavioral model for φ on $s \in \mathrm{St}$, in symbols $\mathcal{G}, s \models_{\mathrm{B}} \varphi$, if $\mathcal{G}, \varnothing, s \models_{\mathrm{B}} \varphi$. An SL[NG] sentence φ is behavioral satisfiable if there is an behavioral 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 behavioral equivalent and vice versa. Thus, we have to be careful when we use natural transformation between formulas because they can preserve the meaning only under the classic semantics. An example of this problem can arise when one want to put a formula in *pnf*. Given two SL[NG] formulas φ_1 and φ_2 with $\mathsf{free}(\varphi_1) = \mathsf{free}(\varphi_2)$, we say that φ_1 behaviorally implies φ_2, in symbols $\varphi_1 \Rightarrow_{\mathrm{E}} \varphi_2$, if for all CGSs $\mathcal{G}$, states $s \in \mathrm{St}$, and $\mathsf{free}(\varphi_1)$-defined s-total assignments $\chi \in \mathrm{Asg}(\mathsf{free}(\varphi_1), s)$, it holds that if $\mathcal{G}, \chi, s \models_{\mathrm{B}} \varphi_1$, then $\mathcal{G}, \chi, s \models_{\mathrm{B}} \varphi_2$. Accordingly, we say that φ_1 is behaviorally 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. Behavioral and Nonbehavioral Semantics

Finally, we show that the introduced concept of behavioral 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 4.12 (SL[NG, 0-alt] *Behavioral*). It is interesting to observe that, for every CGS $\mathcal{G}$ and SL[NG, 0-alt] sentence φ, it holds that $\mathcal{G} \models \varphi$ if and only if $\mathcal{G} \models_{\mathrm{B}} \varphi$. 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 Skolem dependence functions on strategies $\theta \in \mathrm{SDF}_{\mathrm{Str}(s_0)}(\wp)$ are necessarily behavioral.

By Corollary 4.6 of SL strategy quantification, it is easy to see that the following coherence property about the behavioral of the SL[NG] satisfiability holds. Intuitively, it asserts that every behaviorally satisfiable sentence in *pnf* is satisfiable too.

THEOREM 4.13 (SL[NG] BEHAVIORAL 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 $\mathsf{free}(\varphi) \subseteq \mathsf{dom}(\chi)$. Then, it holds that $\mathcal{G}, \chi, s \models_{\mathrm{B}} \varphi$ 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 $\phi \in \mathsf{psnt}(\varphi)$, that is, when ϕ is of the form $\wp\psi$, where $\wp \in \mathrm{Qnt}(\mathsf{free}(\psi))$ is a quantification prefix, and ψ is an agent-closed formula.

Suppose that $\mathcal{G}, \varnothing, s \models_{\mathrm{B}} \wp\psi$. Then, by Definition 4.10 of SL [NG] behavioral semantics, there is a behavioral Skolem dependence function $\theta \in \mathrm{BSDF}_{\mathrm{Str}(s)}(\wp)$ such that for all assignments $\chi \in \mathrm{Asg}([\![\wp]\!], s)$, it holds that $\mathcal{G}, \theta(\chi), s \models_{\mathrm{B}} \psi$. Now, by the inductive hypothesis, there is a Skolem dependence function $\theta \in \mathrm{SDF}_{\mathrm{Str}(s)}(\wp)$ such that for all assignments $\chi \in \mathrm{Asg}([\![\wp]\!], s)$, it holds that $\mathcal{G}, \theta(\chi), s \models \psi$. Hence, by Corollary 4.6 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, that is, there are sentences in *pnf* that are satisfiable but not behavioral satisfiable. Note that the following results already holds for CHP-SL.

THEOREM 4.14 (SL[BG] NONBEHAVIORAL). *There exists a satisfiable* SL[BG, 1-ag, 2-var, 1-alt] *sentence in pnf that is not behaviorally satisfiable.*

PROOF. Consider the 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 $\wp \triangleq [[\mathsf{x}]]\langle\langle \mathsf{y}\rangle\rangle$, $\psi_1 \triangleq (\alpha, \mathsf{x})\mathsf{X}\,\mathsf{p} \leftrightarrow (\alpha, \mathsf{y})\mathsf{X}\,\neg\mathsf{p}$, and $\psi_2 \triangleq (\alpha, \mathsf{x})\mathsf{X}\,\mathsf{X}\,\mathsf{p} \leftrightarrow (\alpha, \mathsf{y})\mathsf{X}\,\mathsf{X}\,\mathsf{p}$, and $\varphi_2 \triangleq [[\mathsf{x}]](\alpha, \mathsf{x})\mathsf{X}\,((\langle\langle \mathsf{x}\rangle\rangle(\alpha, \mathsf{x})\mathsf{X}\,\mathsf{p}) \wedge (\langle\langle \mathsf{x}\rangle\rangle(\alpha, \mathsf{x})\mathsf{X}\,\neg\mathsf{p}))$. Moreover, note that the SL[1G, 1-ag, 1-var, 0-alt] sentence φ_2 is equivalent to the CTL formula $\mathsf{AX}\,((\mathsf{EX}\,\mathsf{p}) \wedge (\mathsf{EX}\,\neg\mathsf{p}))$. Then, it is easy to see that the turn-based CGS $\mathcal{G}_{Rdc}$ of Figure 6 satisfies φ. Indeed, $\mathcal{G}_{Rdc}, \theta(\chi), \mathsf{s}_0 \models \psi_1 \wedge \psi_2$ for all assignments $\chi \in \mathrm{Asg}(\{\mathsf{x}\}, \mathsf{s}_0)$, where the nonbehavioral Skolem dependence function $\theta \in \mathrm{SDF}_{\mathrm{Str}(\mathsf{s}_0)}(\wp)$ is such that $\theta(\chi)(\mathsf{y})(\mathsf{s}_0) = \neg\chi(\mathsf{x})(\mathsf{s}_0)$ and $\theta(\chi)(\mathsf{y})(\mathsf{s}_0 \cdot s_i) = \chi(\mathsf{x})(\mathsf{s}_0 \cdot s_{1-i})$ for all $i \in \{0, 1\}$.

Now, let $\mathcal{G}$ be a generic CGS. If $\mathcal{G} \not\models \varphi$, by Theorem 4.13 of SL[NG] behavioral coherence, it holds that $\mathcal{G} \not\models_{\mathrm{B}} \varphi$. 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 $\mathcal{G} \not\models_{\mathrm{B}} \varphi$, we show that for all behavioral Skolem dependence functions $\theta \in \mathrm{BSDF}_{\mathrm{Str}(s_0)}(\wp)$, there exists an assignment $\chi \in \mathrm{Asg}(\{\mathsf{x}\}, s_0)$ such that $\mathcal{G}, \theta(\chi), s_0 \not\models_{\mathrm{B}} \psi_1 \wedge \psi_2$. To do this, let us fix a behavioral Skolem dependence function θ and an assignment χ. Also, assume $s_1 \triangleq \tau(s_0, \emptyset[\alpha \mapsto \chi(\mathsf{x})(s_0)])$ and $s_2 \triangleq \tau(s_0, \emptyset[\alpha \mapsto \theta(\chi)(\mathsf{y})(s_0)])$. Now, we distinguish between two cases.

—$\mathsf{p} \in \lambda(s_1)$ if and only if $\mathsf{p} \in \lambda(s_2)$. In this case, we can easily observe that $\mathcal{G}, \theta(\chi), s_0 \not\models \psi_1$ and consequently, by Theorem 4.13, it holds that $\mathcal{G}, \theta(\chi), s_0 \not\models_{\mathrm{B}} \psi_1 \wedge \psi_2$. So, we are done.

—$\mathsf{p} \in \lambda(s_1)$ if and only if $\mathsf{p} \notin \lambda(s_2)$. If $\mathcal{G}, \theta(\chi), s_0 \not\models \psi_2$, then, by Theorem 4.13, it holds that $\mathcal{G}, \theta(\chi), s_0 \not\models_{\mathrm{B}} \psi_1 \wedge \psi_2$. So, we are done. Otherwise, let $s_3 \triangleq \tau(s_1, \emptyset[\alpha \mapsto \chi(\mathsf{x})(s_0 \cdot s_1)])$ and $s_4 \triangleq \tau(s_2, \emptyset[\alpha \mapsto \theta(\chi)(\mathsf{y})(s_0 \cdot s_2)])$. Then, it holds that $\mathsf{p} \in \lambda(s_3)$ if and only if $\mathsf{p} \in \lambda(s_4)$. Now, consider a new assignment $\chi' \in \mathrm{Asg}(\{\mathsf{x}\}, s_0)$ such that $\chi'(x)(s_0 \cdot s_2) = \chi(x)(s_0 \cdot s_2)$ and $\mathsf{p} \in \lambda(s_3')$ if and only if $\mathsf{p} \notin \lambda(s_4)$, where $s_3' \triangleq \tau(s_1, \emptyset[\alpha \mapsto \chi'(\mathsf{x})(s_0 \cdot s_1)])$. 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, because of the behavioral of the Skolem dependence function θ, we have that $\theta(\chi')(\mathsf{y})(s_0 \cdot s_2) = \theta(\chi)(\mathsf{y})(s_0 \cdot s_2)$. Consequently, it holds that $s_4 = \tau(s_2, \emptyset[\alpha \mapsto \theta(\chi')(\mathsf{y})(s_0 \cdot s_2)])$. Thus, $\mathcal{G}, \theta(\chi'), s_0 \not\models \psi_2$, which, by Theorem 4.13, implies that $\mathcal{G}, \theta(\chi'), s_0 \not\models_{\mathrm{B}} \psi_1 \wedge \psi_2$. So, we are done.

Thus, the thesis of the theorem holds. □

It is interesting to note that we currently do not know if such a result can be extended to the simpler GL fragment.

Remark 4.15 (*Kinds of Nonbehavioral*). It is worth remarking that the kind of nonbehavioral of the sentence φ shown in the previous theorem can be called *essential*, that is, it cannot be eliminated because of the fact that φ is satisfiable but not behavioral satisfiable. However, there are different sentences, such as the conjunct φ_1 in φ, having both models on which they are behaviorally satisfiable and models, like the CGS $\mathcal{G}_{Rdc}$, on which they are only nonbehaviorally satisfiable. Such a kind of nonbehavioral can be called *nonessential* because it can be eliminated by a suitable choice of the underlying model. Note that a similar reasoning can be done for the dual concept of behavioral, which we call *essential* if all models satisfying a given sentence behaviorally satisfy it as well.

Before continuing, we want to show why we have redefined the concepts of implication and equivalence in the context of behavioral semantics. Consider the SL[BG, 1-ag, 2-var, 1-alt] sentence φ_1 used in Theorem 4.14 of the SL[BG] nonbehavioral result. It is not hard to see that it is equivalent to the SL[1G, 1-ag, 1-var, 0-alt] $\varphi' \triangleq (\langle\langle \mathsf{x} \rangle\rangle(\alpha, \mathsf{x})\psi_1 \leftrightarrow \langle\langle \mathsf{x} \rangle\rangle(\alpha, \mathsf{x})\psi_2) \wedge (\langle\langle \mathsf{x} \rangle\rangle(\alpha, \mathsf{x})\psi_3 \leftrightarrow \langle\langle \mathsf{x} \rangle\rangle(\alpha, \mathsf{x})\psi_4)$, where $\psi_1 \triangleq \mathsf{X}(\mathsf{p} \wedge \mathsf{X}\,\mathsf{p})$, $\psi_2 \triangleq \mathsf{X}(\neg\mathsf{p} \wedge \mathsf{X}\,\mathsf{p})$, $\psi_3 \triangleq \mathsf{X}(\mathsf{p} \wedge \mathsf{X} \neg\mathsf{p})$, and $\psi_4 \triangleq \mathsf{X}(\neg\mathsf{p} \wedge \mathsf{X} \neg\mathsf{p})$. Note that φ' is in turn equivalent to the CTL* formula $(\mathsf{E}\psi_1 \leftrightarrow \mathsf{E}\psi_2) \wedge (\mathsf{E}\psi_3 \leftrightarrow \mathsf{E}\psi_4)$. However, φ_1 and φ' are not behaviorally equivalent because we have that $\mathcal{G}_{Rdc} \not\models_{\mathrm{B}} \varphi_1$ but $\mathcal{G}_{Rdc} \models_{\mathrm{B}} \varphi'$, where $\mathcal{G}_{Rdc}$ is the CGS of Figure 6.

At this point, we can proceed with the proof of the behavioral satisfiability for SL[1G]. It is important to note that there is no gap, in our knowledge, between the logics that are behaviorally satisfiable and those that are not because the fragment SL[BG, 1-ag, 2-var, 1-alt] used in the previous theorem cannot be further reduced because of the fact that otherwise it collapses into SL[1G]. 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 schemes and match in place of the more usual strategy and play to avoid confusion between the concepts related to a CGS and those related to the tool.

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

An *even position* in $\mathcal{A}$ is a finite nonempty sequence of nodes $\varrho \in {\mathrm{N_e}}^+$ such that $(\varrho)_0 = n_0$ and, for all $i \in [0, |\varrho| - 1[$, there exists a node $n \in \mathrm{N_o}$ 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 nonempty sequence of nodes $\varrho = \varrho' \cdot n \in {\mathrm{N_e}}^+ \cdot \mathrm{N_o}$, with $n \in \mathrm{N_o}$, such that ϱ' is an even position and $(\mathsf{lst}(\varrho'), n) \in E_e$. By $\mathrm{Pos_e}$ and $\mathrm{Pos_o}$ we denote, respectively, the sets of even and odd positions.

An *even* (*odd*, respectively) *scheme* in $\mathcal{A}$ is a function $\mathsf{s_e} : \mathrm{Pos_e} \to \mathrm{N_o}$ ($\mathsf{s_o} : \mathrm{Pos_o} \to \mathrm{N_e}$, respectively) that maps each even (odd, respectively) position to an odd (even, respectively) node in a way that is compatible with the edge relation E_e (E_o, respectively), that is, for all $\varrho \in \mathrm{Pos_e}$ ($\varrho \in \mathrm{Pos_o}$, respectively), it holds that $(\mathsf{lst}(\varrho), \mathsf{s_e}(\varrho)) \in E_e$ ($(\mathsf{lst}(\varrho), \mathsf{s_o}(\varrho)) \in E_o$, respectively). By $\mathrm{Sch_e}$ (resp., $\mathrm{Sch_o}$) we indicate the sets of even (odd, respectively) schemes.

A *match* in $\mathcal{A}$ is an infinite sequence of nodes $\varpi \in {\mathrm{N_e}}^\omega$ such that $(\varpi)_0 = n_0$ and, for all $i \in \mathbb{N}$, there exists a node $n \in \mathrm{N_o}$ 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* $\mathsf{mtc} : \mathrm{Sch_e} \times \mathrm{Sch_o} \to \mathrm{Mtc}$ is a function that, given two schemes $\mathsf{s_e} \in \mathrm{Sch_e}$ and $\mathsf{s_o} \in \mathrm{Sch_o}$, returns the unique match $\varpi = \mathsf{mtc}(\mathsf{s_e}, \mathsf{s_o})$ such that for all $i \in \mathbb{N}$, it holds that $(\varpi)_{i+1} = \mathsf{s_o}((\varpi)_{\leq i} \cdot \mathsf{s_e}((\varpi)_{\leq i}))$.

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

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

As first step toward the proof of the behavioral of SL[1G], 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 Skolem dependence function 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 4.16 (*Dependence-versus-Valuation Game*). 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 dependence-versus-valuation game for $\mathcal{G}$ in s over P with respect to $\wp$, and $\flat$ is the TPG $\mathcal{H}(\mathcal{G}, s, \mathrm{P}, \wp, \flat) \triangleq \langle \mathcal{A}(\mathcal{G}, s, \wp, \flat), \mathrm{P} \rangle$, where the TPA $\mathcal{A}(\mathcal{G}, s, \wp, \flat) \triangleq \langle \mathrm{St}, \mathrm{St} \times \mathrm{SDF}_{\mathrm{Ac}}(\wp), E, s \rangle$ has the edge relations defined as follows:

—$E_e \triangleq \{(t, (t, \theta)) : t \in \mathrm{St} \wedge \theta \in \mathrm{SDF}_{\mathrm{Ac}}(\wp)\}$;
—$E_o \triangleq \{((t, \theta), \tau(t, \theta(\mathsf{v}) \circ \zeta_\flat)) : t \in \mathrm{St} \wedge \theta \in \mathrm{SDF}_{\mathrm{Ac}}(\wp) \wedge \mathsf{v} \in \mathrm{Val}_{\mathrm{Ac}}([\![\wp]\!])\}$.[14]

In the next lemma, we state a fundamental relationship between dependence-versus-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 behavioral proof.

LEMMA 4.17 (DEPENDENCE-VERSUS-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 $\mathcal{H}(\mathcal{G}, s, \mathrm{P}, \wp, \flat)$ *if and only if 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" a behavioral Skolem dependence function on strategies, in the sense that it contains all plays obtainable by complete assignments derived from the evaluation of the aforementioned Skolem dependence function. In our context, this concept is used to summarize all relevant information needed to verify the behavioral satisfiability of a sentence.

Definition 4.18 (*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 with respect to $\wp$ and $\flat$ if there exists a behavioral Skolem dependence function $\theta \in \mathrm{BSDF}_{\mathrm{Str}(s)}(\wp)$ such that for all assignments $\chi \in \mathrm{Asg}([\![\wp]\!], s)$, it holds that $\mathsf{play}(\theta(\chi) \circ \zeta_\flat, s) \in \mathrm{P}$.

In the next lemma, we give the second of the two crucial steps in our behavioral proof. In particular, we are able to show a one-to-one relationship between the winning in the dependence-versus-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 4.19 (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 $\mathcal{H}(\mathcal{G}, s, \mathrm{P}, \wp, \flat)$ if and only if P *is an encasement with respect to $\wp$ and $\flat$;*
(ii) if player odd wins $\mathcal{H}(\mathcal{G}, s, \mathrm{P}, \wp, \flat)$, then P *is not an encasement with respect to $\wp$ and $\flat$;*

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

(iii) if P *is a Borelian set and it is not an encasement with respect to* $\wp$ *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 behavioral 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-versus-valuation game. The idea behind this construction resides in the strong similarity between the statement of Corollary 4.6 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 if and only if "there exists a Skolem dependence function such that for all all assignments, it holds that ...". On the other hand, we say that player even wins a game if and only if "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 behavioral quantification.

THEOREM 4.20 (SL[1G] BEHAVIORAL). *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* $\mathsf{free}(\varphi) \subseteq \mathsf{dom}(\chi)$*. Then, it holds that* $\mathcal{G}, \chi, s \models \varphi$ *if and only if* $\mathcal{G}, \chi, s \models_{\mathrm{B}} \varphi$.

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 $\phi \in \mathsf{psnt}(\varphi)$, that is, when ϕ is of the form $\wp\flat\psi$, where $\wp \in \mathrm{Qnt(V)}$ and $\flat \in \mathrm{Bnd(V)}$ 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 4.13 of SL[NG] behavioral 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 behavioral Skolem dependence functions $\theta \in \mathrm{BSDF}_{\mathrm{Str}(s)}(\overline{\wp})$, there is an assignment $\chi \in \mathrm{Asg}([\![\overline{\wp}]\!], s)$ such that $\mathcal{G}, \theta(\chi) \circ \zeta_{\flat}, s \models \psi$. Indeed, suppose by contradiction that there exists a behavioral Skolem dependence function $\theta \in \mathrm{BSDF}_{\mathrm{Str}(s)}(\overline{\wp})$ such that for all assignments $\chi \in \mathrm{Asg}([\![\overline{\wp}]\!], s)$, it holds that $\mathcal{G}, \theta(\chi) \circ \zeta_{\flat}, s \not\models \psi$, that is, $\mathcal{G}, \theta(\chi) \circ \zeta_{\flat}, s \models \neg\psi$, and so $\mathcal{G}, \theta(\chi), s \models \flat\neg\psi$. Then, by Corollary 4.6 of SL strategy quantification, we have that $\mathcal{G}, \varnothing, s \models \overline{\wp}\flat\neg\psi$, that is, $\mathcal{G}, \varnothing, s \models \neg\wp\flat\psi$, 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 behavioral Skolem dependence functions $\theta \in \mathrm{BSDF}_{\mathrm{Str}(s)}(\overline{\wp})$, there is an assignment $\chi \in \mathrm{Asg}([\![\overline{\wp}]\!], s)$ such that $\mathsf{play}(\theta(\chi) \circ \zeta_{\flat}, s) \notin \mathrm{P}$.

At this point, by Definition 4.18 of encasement, it is clear that P is not an encasement with respect to $\overline{\wp}$ and $\flat$. Moreover, because ψ describes a regular language, the derived set P is Borelian [Perrin and Pin 2004]. Consequently, by Item iii of Lemma 4.19 of encasement characterization, we have that player odd wins the TPG $\mathcal{H}(\mathcal{G}, s, \mathrm{P}, \overline{\wp}, \flat)$. Thus, by Lemma 4.17 of dependence-versus-valuation duality, player even wins the dual TPG $\mathcal{H}(\mathcal{G}, s, \mathrm{Pth}(s)\backslash\mathrm{P}, \wp, \flat)$. Hence, by Item i of Lemma 4.19, we have that $\mathrm{Pth}(s)\backslash\mathrm{P}$ is an encasement with respect to $\wp$ and $\flat$. Finally, again by Definition 4.18, there exists a behavioral Skolem dependence function $\theta \in \mathrm{BSDF}_{\mathrm{Str}(s)}(\wp)$ such that for all assignments $\chi \in \mathrm{Asg}([\![\wp]\!], s)$, it holds that $\mathsf{play}(\theta(\chi) \circ \zeta_{\flat}, s) \in \mathrm{Pth}(s)\backslash\mathrm{P}$.

Now, it is immediate to observe that $\mathrm{Pth}(s)\backslash\mathrm{P} = \{\mathsf{play}(\chi, s) \in \mathrm{Pth}(s) : \chi \in \mathrm{Asg}(\mathrm{Ag}, s) \wedge \mathcal{G}, \chi, s \models \psi\}$. So, by the inductive hypothesis, we have that $\mathrm{Pth}(s)\backslash\mathrm{P} = \{\mathsf{play}(\chi, s) \in \mathrm{Pth}(s) : \chi \in \mathrm{Asg}(\mathrm{Ag}, s) \wedge \mathcal{G}, \chi, s \models_{\mathrm{B}} \psi\}$, from which we derive that there exists a behavioral Skolem dependence function $\theta \in \mathrm{BSDF}_{\mathrm{Str}(s)}(\wp)$ such that for all assignments $\chi \in \mathrm{Asg}([\![\wp]\!], s)$, it holds that $\mathcal{G}, \theta(\chi) \circ \zeta_{\flat}, s \models_{\mathrm{B}} \psi$. Consequently, by Definition 4.10 of SL[NG] behavioral semantics, we have that $\mathcal{G}, \varnothing, s \models_{\mathrm{B}} \wp\flat\psi$. □

As an immediate consequence of the previous theorem, we derive the following fundamental corollary, restricted to SL[1G] sentences.

COROLLARY 4.21 (SL[1G] BEHAVIORAL). *Let $\mathcal{G}$ be a* CGS *and φ an* SL[1G] *sentence. Then, $\mathcal{G} \models \varphi$ if and only if $\mathcal{G} \models_{\mathrm{B}} \varphi$.*

It is worth observing that the behavioral property for SL[1G] is a crucial difference with respect to SL[BG], which allows us to obtain a behavioral 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 nonelementarily 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-theoreticapproach [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, because 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 Section 5.1, we recall the definition of alternating parity tree automata. Then, in Section 5.2, we build an automaton accepting a tree encoding of a CGS if and only if this satisfies the formula of interest, which is used to prove the main result about SL and SL[NG] model checking. Finally, in Section 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 of its 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) is a tuple $\mathcal{A} \triangleq \langle \Sigma, \Delta, \mathrm{Q}, \delta, q_0, \aleph \rangle$, where Σ, Δ, and Q are, respectively, nonempty finite sets of input symbols, directions, and states, $q_0 \in \mathrm{Q}$ is an initial state, $\aleph$ is an acceptance condition to be defined later, and $\delta : \mathrm{Q} \times \Sigma \to \mathrm{B}^+(\Delta \times \mathrm{Q})$ 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 $(d, q) \in \Delta \times \mathrm{Q}$ (a.k.a. moves).

On one side, a nondeterministic tree automaton (NTA) 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 $q \in \mathrm{Q}$ 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) 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 $q \in \mathrm{Q}$ 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 $\mathcal{A} = \langle \Sigma, \Delta, \mathrm{Q}, \delta, q_0, \aleph \rangle$ on a Σ-labeled Δ-tree $\mathcal{T} = \langle \mathrm{T}, \mathsf{v} \rangle$ is a $(\Delta \times \mathrm{Q})$-tree R such that for all nodes $x \in \mathrm{R}$, 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) $y \in \mathrm{T}$ and (ii) there is a set of moves $\mathrm{S} \subseteq \Delta \times \mathrm{Q}$ with $\mathrm{S} \models \delta(q_n, \mathsf{v}(y))$ such that $x \cdot (d, q) \in \mathrm{R}$ for all $(d, q) \in \mathrm{S}$.

In the following, we consider ATAs along with the parity acceptance condition (APT) $\aleph \triangleq (\mathrm{F}_1, \ldots, \mathrm{F}_k) \in (2^{\mathrm{Q}})^+$ with $\mathrm{F}_1 \subseteq \cdots \subseteq \mathrm{F}_k = \mathrm{Q}$ (see Kupferman et al. [2000]).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) 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 $\mathsf{inf}(w) \triangleq \{: m\} q \in \mathrm{Q} | \{i \in \mathbb{N} : \exists d \in \Delta.(w)_i = (d, q)\}| = \omega$, 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 $\aleph = (\mathrm{F}_1, \ldots, \mathrm{F}_k)$ if the least index $i \in [1, k]$ for which $\mathsf{inf}(w) \cap \mathrm{F}_i \neq \emptyset$ is even.

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

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

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

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 $f \in \mathrm{B}(\mathrm{P})$ 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 $m \cdot 2^{\mathcal{O}(k \cdot n \cdot \log n)}$ states and index $\mathcal{O}(k \cdot n \cdot \log n)$ such that for all Σ-labeled Δ-tree $\mathcal{T} \triangleq \langle \mathrm{T}, \mathsf{v} \rangle$, it holds that $\mathcal{T} \in \mathrm{L}(\mathcal{N}^{d_0})$ if and only if $\mathcal{T}' \in \mathrm{L}(\mathcal{A})$, 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] 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 $2^{\mathcal{O}(k \cdot n \cdot \log n)}$ states and index $k' = \mathcal{O}(k \cdot n \cdot \log n)$. 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 $m \cdot 2^{\mathcal{O}(k \cdot n \cdot \log n)}$ 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 toward 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 $\mathcal{A}^{\mathcal{G}}_{\varphi}$ 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, if and only if $\mathcal{G}$ 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 $\mathsf{u}(t) \triangleq (\widehat{\chi}(s \cdot t), \mathsf{last}(s \cdot t))$ 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 $\mathcal{A}^{\mathcal{G}}_{\varphi} \triangleq \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathsf{free}(\varphi)) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\varphi}, \delta_{\varphi}, q_{0\varphi}, \aleph_{\varphi} \rangle$ *such 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 $\mathcal{G}, \chi, s \models \varphi$ if and only if $\mathcal{T} \in \mathrm{L}(\mathcal{A}^{\mathcal{G}}_{\varphi})$, where $\mathcal{T}$ is the assignment-state encoding for χ.*

PROOF. The construction of the APT $\mathcal{A}^{\mathcal{G}}_{\varphi}$ is done recursively on the structure of the formula φ, which without loss of generality is supposed to be in *enf*, by using a variation of the transformation, via alternating tree automata, of the S1S and S*k*S logics into nondeterministic Büchi word and tree automata recognizing all models of the formula of interest [Büchi 1962; Rabin 1969].

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

—If $\varphi \in \mathrm{AP}$, the automaton has to verify if the atomic proposition is locally satisfied. To do this, we set $\mathcal{A}^{\mathcal{G}}_{\varphi} \triangleq \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\emptyset) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \{\varphi\}, \delta_{\varphi}, \varphi, (\{\varphi\}) \rangle$, 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, $\mathcal{A}^{\mathcal{G}}_{\varphi}$ 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 $\mathcal{A}^{\mathcal{G}}_{\varphi'} = \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathsf{free}(\varphi')) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\varphi'}, \delta_{\varphi'}, q_{0\varphi'}, \aleph_{\varphi'} \rangle$ [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 $\mathcal{A}^{\mathcal{G}}_{\varphi_1} = \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathsf{free}(\varphi_1)) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\varphi_1}, \delta_{\varphi_1}, q_{0\varphi_1}, \aleph_{\varphi_1} \rangle$ and $\mathcal{A}^{\mathcal{G}}_{\varphi_2} = \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathsf{free}(\varphi_2)) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\varphi_2}, \delta_{\varphi_2}, q_{0\varphi_2}, \aleph_{\varphi_2} \rangle$ into the automaton $\mathcal{A}^{\mathcal{G}}_{\varphi} \triangleq \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathsf{free}(\varphi)) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\varphi}, \delta_{\varphi}, q_{0\varphi}, \aleph_{\varphi} \rangle$, where the following hold:
  —$\mathrm{Q}_{\varphi} \triangleq \{q_{0\varphi}\} \cup \mathrm{Q}_{\varphi_1} \cup \mathrm{Q}_{\varphi_2}$, with $q_{0\varphi} \notin \mathrm{Q}_{\varphi_1} \cup \mathrm{Q}_{\varphi_2}$;
  —$\delta_{\varphi}(q_{0\varphi}, (\mathsf{v}, s)) \triangleq \delta_{\varphi_1}(q_{0\varphi_1}, (\mathsf{v}_{\restriction \mathsf{free}(\varphi_1)}, s))\ \mathsf{Op}\ \delta_{\varphi_2}(q_{0\varphi_2}, (\mathsf{v}_{\restriction \mathsf{free}(\varphi_2)}, s))$ 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) $\mathrm{F}_{i\varphi} \triangleq \mathrm{F}_{i\varphi_1} \cup \mathrm{F}_{i\varphi_2}$, 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 $\mathcal{A}^\mathcal{G}_{\varphi'} = \langle \mathrm{Val}_{\mathrm{Ac}_\mathcal{G}}(\mathsf{free}(\varphi')) \times \mathrm{St}_\mathcal{G}, \mathrm{St}_\mathcal{G}, \mathrm{Q}_{\varphi'}, \delta_{\varphi'}, q_{0\varphi'}, \aleph_{\varphi'}\rangle$ 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 $\mathcal{A}^\mathcal{G}_\varphi \triangleq \langle \mathrm{Val}_{\mathrm{Ac}_\mathcal{G}}(\mathsf{free}(\varphi)) \times \mathrm{St}_\mathcal{G}, \mathrm{St}_\mathcal{G}, \mathrm{Q}_\varphi, \delta_\varphi, q_{0\varphi}, \aleph_\varphi\rangle$, where the following hold:
—$\mathrm{Q}_\varphi \triangleq \{q_{0\varphi}\} \cup \mathrm{Q}_{\varphi'}$, with $q_{0\varphi} \notin \mathrm{Q}_{\varphi'}$;
—$\delta_\varphi(q_{0\varphi},(\mathsf{v},s)) \triangleq (\tau_\mathcal{G}(s,\mathsf{v}_{\restriction \mathrm{Ag}}),q_{0\varphi'})$ for all $(\mathsf{v},s) \in \mathrm{Val}_{\mathrm{Ac}_\mathcal{G}}(\mathsf{free}(\varphi)) \times \mathrm{St}_\mathcal{G}$;
—$\delta_\varphi(q,(\mathsf{v},s)) \triangleq \delta_{\varphi'}(q,(\mathsf{v}_{\restriction \mathsf{free}(\varphi')},s))$ for all $q \in \mathrm{Q}_{\varphi'}$ and $(\mathsf{v},s) \in \mathrm{Val}_{\mathrm{Ac}_\mathcal{G}}(\mathsf{free}(\varphi)) \times \mathrm{St}_\mathcal{G}$;
—$\aleph_\varphi \triangleq (\mathrm{F}_{1\varphi'},\ldots,\mathrm{F}_{k\varphi'} \cup \{q_{0\varphi}\})$, 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 $\mathcal{A}^\mathcal{G}_\varphi \triangleq \langle \mathrm{Val}_{\mathrm{Ac}_\mathcal{G}}(\mathsf{free}(\varphi)) \times \mathrm{St}_\mathcal{G}, \mathrm{St}_\mathcal{G}, \mathrm{Q}_\varphi, \delta_\varphi, q_{0\varphi}, \aleph_\varphi\rangle$ 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 $\mathcal{A}^\mathcal{G}_{\varphi_1} = \langle \mathrm{Val}_{\mathrm{Ac}_\mathcal{G}}(\mathsf{free}(\varphi_1)) \times \mathrm{St}_\mathcal{G}, \mathrm{St}_\mathcal{G}, \mathrm{Q}_{\varphi_1}, \delta_{\varphi_1}, q_{0\varphi_1}, \aleph_{\varphi_1}\rangle$ and $\mathcal{A}^\mathcal{G}_{\varphi_2} = \langle \mathrm{Val}_{\mathrm{Ac}_\mathcal{G}}(\mathsf{free}(\varphi_2)) \times \mathrm{St}_\mathcal{G}, \mathrm{St}_\mathcal{G}, \mathrm{Q}_{\varphi_2}, \delta_{\varphi_2}, q_{0\varphi_2}, \aleph_{\varphi_2}\rangle$ for the inner formulas φ_1 and φ_2. The definitions of $\mathcal{A}^\mathcal{G}_\varphi$ components follows:
—$\mathrm{Q}_\varphi \triangleq \{q_{0\varphi}\} \cup \mathrm{Q}_{\varphi_1} \cup \mathrm{Q}_{\varphi_2}$, with $q_{0\varphi} \notin \mathrm{Q}_{\varphi_1} \cup \mathrm{Q}_{\varphi_2}$;
—$\delta_\varphi(q_{0\varphi},(\mathsf{v},s)) \triangleq \delta_{\varphi_2}(q_{0\varphi_2},(\mathsf{v}_{\restriction \mathsf{free}(\varphi_2)},s)) \vee \delta_{\varphi_1}(q_{0\varphi_1},(\mathsf{v}_{\restriction \mathsf{free}(\varphi_1)},s)) \wedge (\tau_\mathcal{G}(s,\mathsf{v}_{\restriction \mathrm{Ag}}),q_{0\varphi})$ 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)$\mathrm{F}_{i\varphi} \triangleq \{q_{0\varphi}\} \cup \mathrm{F}_{i\varphi_1} \cup \mathrm{F}_{i\varphi_2}$, 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 $\mathrm{F}_{1\varphi}$, 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 $\mathcal{A}^\mathcal{G}_\varphi \triangleq \langle \mathrm{Val}_{\mathrm{Ac}_\mathcal{G}}(\mathsf{free}(\varphi)) \times \mathrm{St}_\mathcal{G}, \mathrm{St}_\mathcal{G}, \mathrm{Q}_\varphi, \delta_\varphi, q_{0\varphi}, \aleph_\varphi\rangle$ 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 $\mathcal{A}^\mathcal{G}_{\varphi_1} = \langle \mathrm{Val}_{\mathrm{Ac}_\mathcal{G}}(\mathsf{free}(\varphi_1)) \times \mathrm{St}_\mathcal{G}, \mathrm{St}_\mathcal{G}, \mathrm{Q}_{\varphi_1}, \delta_{\varphi_1}, q_{0\varphi_1}, \aleph_{\varphi_1}\rangle$ and $\mathcal{A}^\mathcal{G}_{\varphi_2} = \langle \mathrm{Val}_{\mathrm{Ac}_\mathcal{G}}(\mathsf{free}(\varphi_2)) \times \mathrm{St}_\mathcal{G}, \mathrm{St}_\mathcal{G}, \mathrm{Q}_{\varphi_2}, \delta_{\varphi_2}, q_{0\varphi_2}, \aleph_{\varphi_2}\rangle$ for the inner formulas φ_1 and φ_2. The definitions of $\mathcal{A}^\mathcal{G}_\varphi$ 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 $i \in [2,h]$, (v) $\mathrm{F}_{i\varphi} \triangleq \{q_{0\varphi}\} \cup \mathrm{F}_{i\varphi_j}$, for $i \in [h+1,k-1]$ with $k_j = k$, and (vi) $\mathrm{F}_{k\varphi} \triangleq \mathrm{Q}_\varphi$.

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 $\mathrm{F}_{1\varphi}$, to allow its regeneration for an infinite number of time.

—The case $\varphi = \langle\!\langle a, x \rangle\!\rangle \varphi'$ is solved by simply transforming the transition function of the automaton $\mathcal{A}^{\mathcal{G}}_{\varphi'} = \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathsf{free}(\varphi')) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\varphi'}, \delta_{\varphi'}, q_{0\varphi'}, \aleph_{\varphi'} \rangle$, by setting the value of the valuations in input with respect to the agent a to the value of the same valuation with respect to the variable x. The definitions of the transition function for $\mathcal{A}^{\mathcal{G}}_{\varphi} \triangleq \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathsf{free}(\varphi)) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\varphi'}, \delta_{\varphi}, q_{0\varphi'}, \aleph_{\varphi'} \rangle$ 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 $\mathcal{A}^{\mathcal{G}}_{\varphi'}$, by applying the classic transformation [Muller and Schupp 1995]. In this way, we obtain an equivalent NPT $\mathcal{N}^{\mathcal{G}}_{\varphi'} = \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathsf{free}(\varphi')) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\varphi'}, \delta_{\varphi'}, q_{0\varphi'}, \aleph_{\varphi'} \rangle$. Now, we make the projection by defining the new NPT $\mathcal{A}^{\mathcal{G}}_{\varphi} \triangleq \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathsf{free}(\varphi)) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\varphi'}, \delta_{\varphi}, q_{0\varphi'}, \aleph_{\varphi'} \rangle$, 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 $\mathcal{G}, \chi, s \models \varphi$ if and only if $\mathcal{T} \in \mathrm{L}(\mathcal{A}^{\mathcal{G}}_{\varphi})$, 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. □

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 $\mathcal{N}^{\mathcal{G},s}_{\varphi}$ *such that $\mathcal{G}, \varnothing, s \models \varphi$ if and only if $\mathrm{L}(\mathcal{N}^{\mathcal{G},s}_{\varphi}) \neq \emptyset$.*

PROOF. To construct the NPT $\mathcal{N}^{\mathcal{G},s}_{\varphi}$ we apply Theorem 5.4 of APT direction projection with distinguished direction s to the APT $\mathcal{A}^{\mathcal{G}}_{\varphi}$ 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, because φ is a sentence, we have that $\mathsf{free}(\varphi) = \emptyset$, 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 $\mathcal{T} \in \mathrm{L}(\mathcal{A}^{\mathcal{G}}_{\varphi})$, where $\mathcal{T}$ is the dependence-state encoding for $\varnothing$. Hence, by Theorem 5.4, it holds that $\mathrm{L}(\mathcal{N}^{\mathcal{G},s}_{\varphi}) \neq \emptyset$.

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

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 to represent 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 *with respect to the size of the model and* NONELEMENTARY *with respect to 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 $m = \mathtt{O}(\mathsf{lng}(\varphi) \cdot \log \mathsf{lng}(\varphi))$. It is well known that the emptiness problem for such a kind of automaton with n states and index h is solvable in time $\mathtt{O}(n^h)$ [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 with repsect to the size of the model and NONELEMENTARY with respect to 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 claim 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 *with respect to the size of the model and* $(k+1)$-EXPTIME *with respect to 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 $m = \mathtt{O}(\mathsf{lng}(\psi) \cdot \log \mathsf{lng}(\psi))$ 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, because 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 with respect to the size of the model and $(k+1)$-EXPTIME with respect to 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 behavioral of Skolem dependence functions 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 blowups.

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* $\chi \in \mathrm{Asg}_{\mathcal{G}}(\mathsf{free}(\flat\psi), s)$*, it holds that* $\mathcal{G}, \chi, s \models \flat\psi$ *if and only if* $\mathcal{T} \in \mathrm{L}(\mathcal{U}_{\flat\psi}^{\mathcal{G}})$*, where* $\mathcal{T}$ *is the assignment-state encoding for* χ.

PROOF. A first step in the construction of the UCT $\mathcal{U}^{\mathcal{G}}_{\flat\psi}$ 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 $\mathrm{L}(\mathcal{U}_{\psi}) = \mathrm{L}(\psi)$, that is, $\mathcal{U}_{\psi}$ recognizes all infinite words on the alphabet 2^{AP} that satisfy the LTL formula ψ. Then, define the components of $\mathcal{U}^{\mathcal{G}}_{\flat\psi} \triangleq \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathsf{free}(\flat\psi)) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\flat\psi}, \delta_{\flat\psi}, q_{0\flat\psi}, \aleph_{\flat\psi}\rangle$ 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 $(\mathsf{v}, s) \in \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathsf{free}(\flat\psi)) \times \mathrm{St}_{\mathcal{G}}$;
—$\delta_{\flat\psi}(q, (\mathsf{v}, s)) \triangleq \bigwedge_{q'\in\delta_{\psi}(q,\lambda_{\mathcal{G}}(s))}(\tau_{\mathcal{G}}(s, \mathsf{v}\circ\zeta_{\flat}), q')$ for all $q \in \mathrm{Q}_{\psi}$ and $(\mathsf{v}, s) \in \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathsf{free}(\flat\psi)) \times \mathrm{St}_{\mathcal{G}}$;
—$\aleph_{\flat\psi} \triangleq \aleph_{\psi}$.

Intuitively, the UCT $\mathcal{U}^{\mathcal{G}}_{\flat\psi}$ 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 $\chi \in \mathrm{Asg}_{\mathcal{G}}(\mathsf{free}(\flat\psi), s)$, it holds that $\mathcal{G}, \chi, s \models \flat\psi$ if and only if $\mathcal{T} \in \mathrm{L}(\mathcal{U}^{\mathcal{G}}_{\flat\psi})$, 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 behavioral Skolem dependence functions over strategies.

Definition 5.11 (*Behavioral Dependence-State Encoding*). Let $\mathcal{G}$ be a CGS, $s \in \mathrm{St}_{\mathcal{G}}$ one of its states, and $\theta \in \mathrm{BSDF}_{\mathrm{Str}_{\mathcal{G}}(s)}(\wp)$ a behavioral Skolem dependence function over strategies for a quantification prefix $\wp \in \mathrm{Qnt}(\mathrm{V})$ over the set $\mathrm{V} \subseteq \mathrm{Var}$. Then, a $(\mathrm{SDF}_{\mathrm{Ac}_{\mathcal{G}}}(\wp) \times \mathrm{St}_{\mathcal{G}})$-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 a behavioral 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 behavioral dependence-state encoding for each behavioral Skolem dependence function 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 Skolem dependence function. Intuitively, we guess in the labeling what is the right part of the Skolem dependence function 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 $\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi} \triangleq \langle \mathrm{SDF}_{\mathrm{Ac}_{\mathcal{G}}}(\wp) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\wp\flat\psi}, \delta_{\wp\flat\psi}, q_{0\wp\flat\psi}, \aleph_{\wp\flat\psi}\rangle$ *such that for all states $s \in \mathrm{St}_{\mathcal{G}}$ and behavioral Skolem dependence functions over strategies $\theta \in \mathrm{BSDF}_{\mathrm{Str}_{\mathcal{G}}(s)}(\wp)$, it holds that $\mathcal{G}, \theta(\chi), s \models_{\mathrm{B}} \flat\psi$ for all $\chi \in \mathrm{Asg}_{\mathcal{G}}([\![\wp]\!], s)$ if and only if $\mathcal{T} \in \mathrm{L}(\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi})$, where $\mathcal{T}$ is the behavioral dependence-state encoding for θ.*

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

Now, transform $\mathcal{U}^{\mathcal{G}}_{\flat\psi}$ into the new UCT $\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi} \triangleq \langle \mathrm{SDF}_{\mathrm{Ac}_{\mathcal{G}}}(\wp) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\wp\flat\psi}, \delta_{\wp\flat\psi}, q_{0\wp\flat\psi}, \aleph_{\wp\flat\psi}\rangle$, 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:

$\delta_{\wp\flat\psi}(q,(\theta,s)) \triangleq \bigwedge_{\mathsf{v}\in \mathrm{Val}_{\mathrm{Ac}_\mathcal{G}}([\![\wp]\!])} \delta_{\flat\psi}(q,(\theta(\mathsf{v}),s))$ for all $q \in \mathrm{Q}_{\wp\flat\psi}$ and $(\theta,s) \in \mathrm{SDF}_{\mathrm{Ac}_\mathcal{G}}(\wp) \times \mathrm{St}_\mathcal{G}$. Intuitively, $\mathcal{U}^\mathcal{G}_{\wp\flat\psi}$ reads an action Skolem dependence function 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 $\delta_{\flat\psi}(q,(\mathsf{v},s))$ of $\mathcal{U}^\mathcal{G}_{\flat\psi}$ for each possible valuation $\mathsf{v} = \theta(\mathsf{v}')$ on $\mathsf{free}(\flat\psi)$ obtained from θ by a universal valuation $\mathsf{v}' \in \mathrm{Val}_{\mathrm{Ac}_\mathcal{G}}([\![\wp]\!])$. It is important to observe that we cannot move the component set $\mathrm{SDF}_{\mathrm{Ac}_\mathcal{G}}(\wp)$ from the input alphabet to the states of $\mathcal{U}^\mathcal{G}_{\wp\flat\psi}$ by making a related guessing of the Skolem dependence function θ in the transition function because we have to ensure that all states in a given node of the tree $\mathcal{T}$, that is, 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 behavioral Skolem dependence function over strategies $\theta \in \mathrm{BSDF}_{\mathrm{Str}_\mathcal{G}(s)}(\wp)$, it holds that $\mathcal{G},\theta(\chi),s \models_\mathrm{B} \flat\psi$ for all $\chi \in \mathrm{Asg}_\mathcal{G}([\![\wp]\!],s)$ if and only if $\mathcal{T} \in \mathrm{L}(\mathcal{U}^\mathcal{G}_{\wp\flat\psi})$, where $\mathcal{T}$ is the behavioral dependence-state encoding for θ.

[Only if]. Suppose that $\mathcal{G},\theta(\chi),s \models_\mathrm{B} \flat\psi$ for all $\chi \in \mathrm{Asg}_\mathcal{G}([\![\wp]\!],s)$. Because ψ does not contain principal subsentences, we have that $\mathcal{G},\theta(\chi),s \models \flat\psi$. So, because of 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, because of 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 $\mathcal{U}_{\flat\psi|^\mathcal{G}}$ on the assignment-state encoding $\mathcal{T}_\chi$ for $\theta(\chi)$. Because of 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, because 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, because of the property of $\mathcal{U}^\mathcal{G}_{\flat\psi}$, it follows that $\mathcal{G},\theta(\chi),s \models \flat\psi$. Now, because ψ does not contain principal subsentences, we have that $\mathcal{G},\theta(\chi),s \models_\mathrm{B} \flat\psi$ for all $\chi \in \mathrm{Asg}_\mathcal{G}([\![\wp]\!],s)$. □

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* $\mathcal{G},\varnothing,s \models \wp\flat\psi$ *if and only if* $\mathrm{L}(\mathcal{N}^{\mathcal{G},s}_{\wp\flat\psi}) \neq \emptyset$.

Proof. As in the general case of Sl sentence automaton, we have to ensure that the state labeling of nodes of the behavioral 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 4.21 of Sl[1g] behavioral, it means that $\mathcal{G},\varnothing,s \models_\mathrm{B} \wp\flat\psi$. Then, by Definition 4.10 of Sl[ng] behavioral semantics, there exists an behavioral Skolem dependence function $\theta \in \mathrm{BSDF}_{\mathrm{Str}_\mathcal{G}(s)}(\wp)$ such that $\mathcal{G},\theta(\chi),s \models_\mathrm{B} \flat\psi$ 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 behavioral 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 $(\mathrm{SDF}_{\mathrm{Ac}_\mathcal{G}}(\wp) \times \mathrm{St}_\mathcal{G})$-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 a behavioral Skolem dependence function $\theta \in \mathrm{BSDF}_{\mathrm{Str}_\mathcal{G}(s)}(\wp)$ for which $\mathcal{T}$ is the behavioral dependence-state encoding. Thus, by Lemma 5.12, we have that $\mathcal{G}, \theta(\chi), s \models_{\mathrm{B}} \flat\psi$ for all $\chi \in \mathrm{Asg}_\mathcal{G}([\![\wp]\!], s)$. By Definition 4.10 of SL[NG] behavioral semantics, it holds that $\mathcal{G}, \varnothing, s \models_{\mathrm{B}} \wp\flat\psi$. Hence, by Corollary 4.21 of SL[1G] behavioral, 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 *with respect to the size of the model and* 2 EXPTIME-COMPLETE *with respect to 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 $2^{2^{\mathtt{O}(\mathsf{lng}(\psi))}}$. So, $\mathcal{N}^{\mathcal{G},s}_{\wp\flat\psi}$ has $|\mathrm{St}_\mathcal{G}| \cdot 2^{2^{\mathtt{O}(\mathsf{lng}(\psi))}}$ states and index $2^{\mathtt{O}(\mathsf{lng}(\psi))}$.

The emptiness problem for such a kind of automaton with n states and index h is solvable in time $\mathtt{O}(n^h)$ [Kupferman and Vardi 1998]. Thus, we get that the time complexity of checking whether $\mathcal{G}, \varnothing, s \models \wp\flat\psi$ is $|\mathrm{St}_\mathcal{G}|^{2^{\mathtt{O}(\mathsf{lng}(\psi))}}$. At this point, because 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 $|\mathrm{St}_\mathcal{G}|^{2^{\mathtt{O}(\mathsf{lng}(\varphi))}}$. Hence, the membership of the model-checking problem for SL[1G] in PTIME with respect to the size of the model and 2EXPTIME with repsect to 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

Finding the right logic for reasoning about games is an important and intriguing task. Current proposals for infinite games on state-transition systems follow two main directions, either extending first-order temporal logics with a mechanism for quantifying over plays built by two alternating (coalitions of) players, or restricting monadic second-order logic (MSOL) to speak about infinite trees with respect to outcomes of the strategies they encode rather than about their internal structure. Both approaches are unsatisfactory: alternating-time logics are computationally acceptable, but they lack expressiveness, whereas MSOL or μ-calculus–based formalisms tend to be expressive, but of nonelementary complexity, because of the unbounded nesting of quantifiers. This article combines the two approaches leading to a logic that is strongly expressive and overly complex in its full version, but which admits a fragment of acceptable computational complexity that is expressive enough to counter some of the criticisms brought to alternating-time logics.

Specifically, in this article, we have introduced and studied SL as a very powerful logic formalism to reasoning about strategic behaviors of multiagent concurrent games. In particular, we have proved that this new logic subsumes the classical temporal and game logics not using explicit fix-points. As one of the main results about SL, we have shown that the relative model-checking problem is decidable but nonelementary hard. As further and interesting practical results, we have 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 article, we show 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 *behavioral*. 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. Thus, 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 multiagent concurrent systems.

As another interesting fragment we investigated in this article, 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 conjecture that although it does not have the behavioral property, its model-checking problem has an elementary complexity. On the other hand, Mogavero et al. [2013] have proven that if the logic fragment having only conjunctions of goals has the behavioral property, then its model-checking problem is 2EXPTIME-COMPLETE.

An important aspect of the logic SL is that its ability in the strategic reasoning is independent from the temporal logic setting used to specify goals, but rather intrinsic in the skeleton of the logic itself. To give evidence of this claim, we report that Mogavero et al. [2014b] proposed a revisit of this logic under a game-theoretic framework. Specifically, the new formalization has been obtained by adopting the general guidelines of game-theory, which requires the definition of solution concepts (i.e., the pure strategic reasoning) to abstract away the underlying cost-benefit analysis (together with the temporal constraints used to determine it). The proposed framework, besides gaining in modularity and applicability, shows that all principal concepts and properties related to SL (including the behavioral property) are independent from the specific formalism used to express goals.

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 multiagent 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 interesting 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, 2010, 2012], Kupferman et al. [2002], and 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 the graded full computation tree logic (GCTL*) [Bianco et al. 2012].

ELECTRONIC APPENDIX

The electronic appendix for this article can be accessed in the ACM Digital Library.

ACKNOWLEDGMENT

We wish to thank Da Costa et al. [2010a] for their helpful comments and discussions on a preliminary version of this article.

REFERENCES

T. Agotnes, V. Goranko, and W. Jamroga. 2007. Alternating-time temporal logics with irrevocable strategies. In *TARK*. 15–24.

M. Aigner and M. Fromme. 1984. A game of cops and robbers. *Discrete Appl. Math.* 8, 1 (1984), 1–12.

R. Alur, T. A. Henzinger, and O. Kupferman. 2002. Alternating-time temporal logic. *J. ACM* 49, 5 (2002), 672–713.

B. Aminof, A. Legay, A. Murano, O. Serre, and M. Y. Vardi. 2013. Pushdown module checking with imperfect information. *Inf. Comput.* 223 (2013), 1–17.

F. Belardinelli. 2014. Reasoning about knowledge and strategies: Epistemic strategy logic. In *SR*. 27–33.

M. Benerecetti, F. Mogavero, and A. Murano. 2013. Substructure temporal logic. In *IEEE Symposium on Logic in Computer Science'13*. IEEE Computer Society, 368–377.

A. Bianco, F. Mogavero, and A. Murano. 2009. Graded computation tree logic. In *IEEE Symposium on Logic in Computer Science'09*. IEEE Computer Society, 342–351.

A. Bianco, F. Mogavero, and A. Murano. 2010. Graded computation tree logic with binary coding. In *EACSL Annual Conference on Computer Science Logic'10*. Springer, 125–139.

A. Bianco, F. Mogavero, and A. Murano. 2012. Graded computation tree logic. *ACM Trans. Comput. Logic* 13, 3 (2012).

P. A. Bonatti, C. Lutz, A. Murano, and M. Y. Vardi. 2008. The complexity of enriched mu-calculi. *Logical Methods Comput. Sci.* 4, 3 (2008), 1–27.

T. Brihaye, A. D. C. Lopes, F. Laroussinie, and N. Markey. 2009. ATL with strategy contexts and bounded memory. In *Symposium on Logical Foundations of Computer Science'09*. Springer, 92–106.

J. R. Büchi. 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.

N. Bulling and W. Jamroga. 2014. Comparing variants of strategic ability: How uncertainty and memory influence general properties of games. *Auton. Agents Multi-Agent Syst.* 28, 3 (2014), 474–518.

N. Bulling, W. Jamroga, and J. Dix. 2008. Reasoning about temporal properties of rational play. *Ann. Math. Artif. Intell.* 53, 1–4 (2008), 51–114.

P. Čermák, A. Lomuscio, F. Mogavero, and A. Murano. 2014. MCMAS-SLK: A model checker for the verification of strategy logic specifications. In *Computer Aided Verification'14*. Springer, 524–531.

C. Chareton, J. Brunel, and D. Chemouil. 2013. Towards an updatable strategy logic. In *Proceedings 1st International Workshop on Strategic Reasoning (SR'13)*, F. Mogavero, A. Murano, and M. Y. Vardi (Eds.). 91–98.

K. Chatterjee, L. Doyen, E. Filiot, and J. F. Raskin. 2014. Doomsday equilibria for omega-regular games. In *VMCAI 2014*. Springer, 78–97.

K. Chatterjee, T. A. Henzinger, and N. Piterman. 2007. Strategy logic. In *International Conference on Concurrency Theory'07*. Springer, 59–73.

K. Chatterjee, T. A. Henzinger, and N. Piterman. 2010. Strategy logic. *Information and Computation* 208, 6 (2010), 677–693.

E. M. Clarke and E. A. Emerson. 1981. Design and synthesis of synchronization skeletons using branching-time temporal logic. In *Logic of Programs'81*. Springer, 52–71.

E. M. Clarke, O. Grumberg, and D. A. Peled. 2002. *Model Checking*. MIT Press.

A. Da Costa, F. Laroussinie, and N. Markey. 2010a. ATL with strategy contexts: Expressiveness and model checking. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science'10*. 120–132.

A. Da Costa, F. Laroussinie, and N. Markey. 2010b. Personal communication.

A. Da Costa, F. Laroussinie, and N. Markey. 2012. Quantified CTL: Expressiveness and model checking (extended abstract). In *CONCUR*. Lecture Notes in Computer Science, Vol. 7454. Springer, 177–192.

H. D. Ebbinghaus and J. Flum. 1995. *Finite Model Theory.* Springer-Verlag.

E. A. Emerson. 1990. Temporal and modal logic. In *Handbook of Theoretical Computer Science*, vol. B. MIT Press, 995–1072.

E. A. Emerson and J. Y. Halpern. 1986. "Sometimes" and "not never" revisited: On branching versus linear time. *J. ACM* 33, 1 (1986), 151–178.

R. Fagin, J. Y. Halpern, Y. Moses, and M. Y. Vardi. 1995. *Reasoning About Knowledge.* MIT Press.

A. Ferrante, A. Murano, and M. Parente. 2008. Enriched μ-calculi module checking. *Logical Methods Comput. Sci.* 4, 3 (2008).

B. Finkbeiner and S. Schewe. 2010. Coordination logic. In *EACSL Annual Conference on Computer Science Logic'10.* Springer, 305–319.

D. Fisman, O. Kupferman, and Y. Lustig. 2010. Rational synthesis. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems'10.* Springer, 190–204.

E. Grädel, W. Thomas, and T. Wilke. 2002. *Automata, Logics, and Infinite Games: A Guide to Current Research.* Springer-Verlag.

A. Herzig, E. Lorini, and D. Walther. 2013. Reasoning about actions meets strategic logics. In *Logic, Rationality, and Interaction*. Springer, 162–175.

W. Hodges. 1993. *Model Theory.* Cambridge University Press.

W. Hodges. 2001. *Elementary Predicate Logic.* Handbook of Philosophical Logic, Vol. 1. Springer.

C. Huang, S. Schewe, and F. Wang. 2013. Model-Checking iterated games. In *Tools and Algorithms for the Construction and Analysis of Systems*. Springer, 154–168.

W. Jamroga and A. Murano. 2014. On module checking and strategies. In *AAMAS*. IFAAMAS, 701–708.

W. Jamroga and W. van der Hoek. 2004. Agents that know how to play. *Fundamenta Informaticae* 63, 2–3 (2004), 185–219.

R. M. Keller. 1976. Formal verification of parallel programs. *Commun. ACM* 19, 7 (1976), 371–384.

F. Kolaitis. 1985. Game quantification. In *Handbook of Model-Theoretic Logics*, J. Barwise and S. Feferman (Eds.). SpringerVerlag, 365–421.

D. Kozen. 1983. Results on the propositional mu-calculus. *Theoretical Comput. Sci.* 27, 3 (1983), 333–354.

S. A. Kripke. 1963. Semantical considerations on modal logic. *Acta Philosophica Fennica* 16 (1963), 83–94.

O. Kupferman, U. Sattler, and M. Y. Vardi. 2002. The complexity of the graded μ-calculus. In *Conference on Automated Deduction'02.* Springer, 423–437.

O. Kupferman and M. Y. Vardi. 1998. Weak alternating automata and tree automata emptiness. In *ACM Symposium on Theory of Computing'98*. 224–233.

O. Kupferman, M. Y. Vardi, and P. Wolper. 2000. An automata theoretic approach to branching-time model checking. *J. ACM* 47, 2 (2000), 312–360.

O. Kupferman, M. Y. Vardi, and P. Wolper. 2001. Module checking. *Inf. Comput.* 164, 2 (2001), 322–344.

F. Laroussinie and N. Markey. 2013. Satisfiability of ATL with strategy contexts. In *GandALF (EPTCS)*, Vol. 119. 208–223.

A. D. Martin. 1975. Borel determinacy. *Ann. Math.* 102, 2 (1975), 363–371.

A. D. Martin. 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.

F. Mogavero, A. Murano, G. Perelli, and M. Y. Vardi. 2012. What makes ATL* decidable? A decidable fragment of strategy logic. In *CONCUR*. 193–208.

F. Mogavero, A. Murano, and L. Sauro. 2013. On the boundary of behavioral strategies. In *IEEE Symposium on Logic in Computer Science'13*. 263–272.

F. Mogavero, A. Murano, and L. Sauro. 2014a. A behavioral hierarchy of strategy logic. In *CLIMA XV.* Springer 148–165.

F. Mogavero, A. Murano, and L. Sauro. 2014b. Strategy games: A renewed framework. In *AAMAS*. IFAAMAS, 869–876.

F. Mogavero, A. Murano, and M. Y. Vardi. 2010a. Reasoning about strategies. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science'10.* 133–144.

F. Mogavero, A. Murano, and M. Y. Vardi. 2010b. Relentful strategic reasoning in alternating-time temporal logic. In *International Conference on Logic for Programming Artificial Intelligence and Reasoning'10.* Springer, 371–387.

D. E. Muller, A. Saoudi, and P. E. Schupp. 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.

D. E. Muller and P. E. Schupp. 1987. Alternating automata on infinite trees. *Theoretical Computer Science* 54, 2–3 (1987), 267–276.

D. E. Muller and P. E. Schupp. 1995. Simulating alternating tree automata by nondeterministic automata: New results and new proofs of theorems of rabin, mcnaughton, and safra. *Theor. Comput. Sci.* 141, 1–2 (1995), 69–107.

R. B. Myerson. 1997. *Game Theory: Analysis of Conflict*. Harvard University Press.

M. Pauly. 2002. A modal logic for coalitional power in games. *J. Logic Comput.* 12, 1 (2002), 149–166.

D. Perrin and J. Pin. 2004. *Infinite Words.* Pure and Applied Mathematics, Vol. 141. Elsevier.

S. Pinchinat. 2007. A generic constructive solution for concurrent games with expressive constraints on strategies. In *International Symposium on Automated Technology for Verification and Analysis'07*. Springer, 253–267.

A. Pnueli. 1977. The temporal logic of programs. In *Foundation of Computer Science'*77. 46–57.

J. P. Queille and J. Sifakis. 1981. Specification and verification of concurrent programs in CESAR. In *International Symposium on Programming'81.* Springer, 337–351.

M. O. Rabin. 1969. Decidability of second-order theories and automata on infinite trees. *Trans. Am. Math. Soc.* 141 (1969), 1–35.

S. Schewe. 2008. ATL* satisfiability is 2exptime-complete. In *International Colloquium on Automata, Languages and Programming'08.* Springer, 373–385.

A. P. Sistla. 1983. *Theoretical Issues in the Design and Certification of Distributed Systems.* Ph.D. Dissertation. Harvard University, Cambridge, MA.

A. P. Sistla, M. Y. Vardi, and P. Wolper. 1987. The complementation problem for Büchi automata with applications to temporal logic. *Theor. Comput. Sci.* 49 (1987), 217–237.

W. Thomas. 1990. Automata on infinite objects. In *Handbook of Theoretical Computer Science*, vol. B. MIT Press, 133–191.

M. Y. Vardi. 1988. A temporal fixpoint calculus. In *ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages'88*. 250–259.

M. Y. Vardi. 1996. Why is modal logic so robustly decidable? In *Descriptive Complexity and Finite Models'96*. American Mathematical Society, 149–184.

M. Y. Vardi and P. Wolper. 1986. An automata-theoretic approach to automatic program verification. In *IEEE Symposium on Logic in Computer Science'86*. IEEE Computer Society, 332–344.

D. Walther, W. van der Hoek, and M. Wooldridge. 2007. Alternating-time temporal logic with explicit strategies. In *TARK*. 269–278.

F. Wang, C. Huang, and F. Yu. 2011. A temporal logic for the interaction of strategies. In *CONCUR 2011*. Springer, 466–481.

Received May 2012; revised March 2014; accepted June 2014