What Makes ATL* Decidable? A Decidable Fragment of Strategy Logic

Fabio Mogavero[1⋆], Aniello Murano[1⋆⋆], Giuseppe Perelli[1], and Moshe Y. Vardi[2⋆⋆⋆]

[1]Università degli Studi di Napoli "Federico II", Napoli, Italy. [2]Rice University, Houston, TX-USA.
{mogavero, murano}@na.infn.it perelli.gi@gmail.com vardi@cs.rice.edu

Abstract *Strategy Logic* (SL, for short) has been recently introduced by Mogavero, Murano, and Vardi as a formalism for reasoning explicitly about strategies, as first-order objects, in multi-agent concurrent games. This logic turns out to be very powerful, strictly subsuming all major previously studied modal logics for strategic reasoning, including ATL, ATL*, and the like. The price that one has to pay for the expressiveness of SL is the lack of important model-theoretic properties and an increased complexity of decision problems. In particular, SL does not have the bounded-tree model property and the related satisfiability problem is *highly undecidable* while for ATL* it is 2EXPTIME-COMPLETE. An obvious question that arises is then what makes ATL* decidable. Understanding this should enable us to identify decidable fragments of SL. We focus, in this work, on the limitation of ATL* to allow only one temporal goal for each strategic assertion and study the fragment of SL with the same restriction. Specifically, we introduce and study the syntactic fragment *One-Goal Strategy Logic* (SL[1G], for short), which consists of formulas in prenex normal form having a single temporal goal at a time for every strategy quantification of agents. We show that SL[1G] is strictly more expressive than ATL*. Our main result is that SL[1G] has the bounded tree-model property and its satisfiability problem is 2EXPTIME-COMPLETE, as it is for ATL*.

1 Introduction

In open-system verification [4, 14], an important area of research is the study of modal logics for strategic reasoning in the setting of multi-agent games [2, 11, 22]. An important contribution in this field has been the development of *Alternating-Time Temporal Logic* (ATL*, for short), introduced by Alur, Henzinger, and Kupferman [2]. ATL* allows reasoning about strategic behavior of agents with temporal goals. Formally, it is obtained as a generalization of the branching-time temporal logic CTL* [6], where 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 "$[\![\mathrm{A}]\!]$", for a set A of *agents*. Such strategic modalities are used to express cooperation and competition among agents in order to achieve certain temporal goals. In particular, these modalities express selective quantifications over those paths that are the results of infinite games between a coalition and its complement. ATL* formulas

⋆ Part of this research was done while visiting the Rice University.
⋆⋆ Work supported in part by University of Naples Federico II under the F.A.R.O. project.
⋆⋆⋆ Work supported in part by NSF grants CNS 1049862 and CCF-1139011, by BSF grant 9800096, and by gift from Intel.

are interpreted over *concurrent game structures* (CGS, for short) [2], which model interacting processes. Given a CGS $\mathcal{G}$ and a set A of agents, the ATL* formula $\langle\langle \mathrm{A} \rangle\rangle \psi$ holds at a state s of $\mathcal{G}$ if there is a set of strategies for the agents in A such that, no matter which strategy is executed by the agents not in A, the resulting outcome of the interaction in $\mathcal{G}$ satisfies ψ at s. Several decision problems have been investigated about ATL*; both its model-checking and satisfiability problems are decidable in 2EXPTIME [26].

Despite its powerful expressiveness, ATL* suffers from the strong limitation that strategies are treated only implicitly through modalities that refer to games between competing coalitions. To overcome this problem, Chatterjee, Henzinger, and Piterman introduced *Strategy Logic* (CHP-SL, for short) [3], a logic that treats strategies in *two-player turn-based games* as *first-order objects*. The explicit treatment of strategies in this logic allows the expression of many properties not expressible in ATL*. Although the model-checking problem of CHP-SL is known to be decidable, with a non-elementary upper bound, it is not known if the satisfiability problem is decidable [3]. While the basic idea exploited in [3] of explicitly quantify over strategies is powerful and useful [8], CHP-SL still suffers from various limitations. In particular, it is limited to two-player turn-based games. Furthermore, CHP-SL does not allow different players to share the same strategy, suggesting that strategies have yet to become truly first-class objects in this logic. For example, it is impossible to describe the classic strategy-stealing argument of combinatorial games such as Hex and the like [1].

These considerations led us to introduce a new *Strategy Logic*, denoted SL, as a more general framework than CHP-SL, for explicit reasoning about strategies in multi-agent concurrent games [18]. Syntactically, SL extends the linear-time temporal-logic LTL [24] by means of *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 read as *"there exists a strategy x"*, *"for all strategies x"*, and *"bind agent a to the strategy associated with x"*, respectively. For example, in a CGS $\mathcal{G}$ with agents α, β, and γ, consider the property "α and β have a common strategy to avoid a failure". This property can be expressed by the SL formula $\langle\langle \mathsf{x} \rangle\rangle [\![\mathsf{y}]\!](\alpha, \mathsf{x})(\beta, \mathsf{x})(\gamma, \mathsf{y})(\mathsf{G}\, \neg \mathit{fail})$. The variable x is used to select a strategy for the agents α and β, while y is used to select another one for agent γ such that their composition, after the binding, results in a play where fail is never met. Additional material can be found in [16].

The price that one has to pay for the expressiveness of SL w.r.t. ATL* is the lack of important model-theoretic properties and an increased complexity of decision problems. In particular, in [18], it was shown that SL does not have the bounded-tree model property and the related satisfiability problem is *highly undecidable*, precisely, Σ^1_1-HARD. Hence, a natural question that arises is what makes ATL* decidable. Understanding the reasons for the decidability of ATL* should enable us to identify decidable fragments of SL.

In this work, we focus on the limitation of ATL* to allow only one temporal goal for each strategic assertion and study the fragment of SL with the same restriction. Specifically, we introduce the syntactic fragment *One-Goal Strategy Logic* (SL[1G], for short), which consists of formulas in a special prenex normal form having a single temporal goal at a time, for every strategy quantification of agents. This means that every temporal formula ψ is prefixed with a quantification-binding prefix that quantifies over a tuple of strategies and bind all agents to strategies. It is worth noting that SL[1G] still

retains the ability to alternate strategy quantifiers as it is for SL, which is not allowed in ATL*. Roughly speaking, SL[1G] is ATL* augmented with this no-limitation on quantifier alternation and with the possibility to force agents to share strategies. This makes SL[1G] strictly more expressive and much more flexible than ATL*, as is shown in [16]. With SL[1G] one can express, for example, visibility constraints on strategies among agents, i.e., only some agents from a coalition have knowledge of the strategies taken by those in the opponent coalition, via the quantifier alternation. Also, by means of strategy sharing, one can describe the fact that, in the Hex game, the strategy-stealing argument does not let the player who adopts it to win. Observe that these properties cannot be expressed neither in ATL* nor in CHP-SL.

In this paper, we show that the satisfiability problem for SL[1G] is also 2EXPTIME-COMPLETE. Thus, in spite of its expressiveness, SL[1G] has the same computational complexities as ATL*. From this result, we conclude that the one-goal restriction is the key aspect to the elementary complexity of ATL*, while the arbitrary quantifier alternation does not let the complexity of the satisfiability problem to rise to non-elementary, as it usually happens in other logics, such as MSOL [25]. In [16], we also introduce SL[NG] and SL[BG] as two additional fragments of SL that strictly include SL[1G]. In SL[NG] we allow writing nesting and boolean combinations of temporal goals, while in SL[BG] we forbid the nesting, but still allow the boolean combinations. Both fragments do not satisfy important model-theoretic properties and have an highly undecidable satisfiability problem. Hence, at the present time, SL[1G] is the most general fragment of SL that subsumes ATL* while keeping its positive model-theoretic and computational properties.

To achieve our main result, we use a fundamental property of the semantics of SL[1G] called *elementariness*, which allows us to simplify reasoning about strategies by reducing it to a set of reasonings about actions. This intrinsic characteristic of SL[1G] means that, to choose an existential strategy, we do not need to know the entire structure of universally-quantified strategies, as it is the case for SL, but only their values on the histories of interest. Technically, to formally describe this property, we make use of the machinery of *dependence maps*, which is introduced to define a Skolemization procedure for SL, inspired by the one in first-order logic. Using elementariness, we show that SL[1G] satisfies the *bounded tree-model property*. This allows us to efficiently make use of a *tree automata-theoretic approach* [27, 29] to solve the satisfiability problem. Given a formula φ, we build an *alternating co-Büchi tree automaton* [13, 21], whose size is only exponential in the size of φ, accepting all bounded-branching tree models of the formula. Then, together with the complexity of automata-nonemptiness checking, we get that the satisfiability procedure for SL[1G] is 2EXPTIME. For completeness, we report that in [16] we already prove that the model-checking problem for SL[1G] remains 2EXPTIME-COMPLETE, while it is non-elementarily decidable for SL.

Related works. Several works have focused on extensions of ATL* to incorporate more powerful strategic constructs. Among them, we recall the *Alternating-Time* μCALCULUS (AμCALCULUS, for short) [2], *Game Logic* (GL, for short) [2], *Quantified Decision Modality* μCALCULUS (QDμ, for short) [23], *Coordination Logic* (CL, for short) [7], and some other extensions considered in [5], [19], and [30]. AμCALCULUS and QDμ are intrinsically different from SL[1G] (as well as from CHP-SL and ATL*) as they are obtained by extending the propositional μ-calculus [12] with strategic modalities. CL is

similar to QDμ, but with LTL temporal operators instead of explicit fixpoint constructors. GL and CHP-SL are orthogonal to SL[1G]. Indeed, they both use more than a temporal goal, GL has quantifier alternation fixed to one, and CHP-SL only works for two agents.

Due to lack of space, proofs are reported in [17]. Also, see [16] for more on SL[1G].

2 Preliminaries

A *concurrent game structure* (CGS, for short) [2] is a tuple $\mathcal{G} \triangleq \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}, \mathrm{St}, \lambda, \tau, s_0 \rangle$, where AP and Ag are finite non-empty sets of *atomic propositions* and *agents*, Ac and St are enumerable non-empty sets of *actions* and *states*, $s_0 \in \mathrm{St}$ is a designated *initial state*, and $\lambda : \mathrm{St} \to 2^{\mathrm{AP}}$ is a *labeling function* that maps each state to the set of atomic propositions true in that state. Let $\mathrm{Dc} \triangleq \mathrm{Ac}^{\mathrm{Ag}}$ be the set of *decisions*, i.e., functions from Ag to Ac representing the choices of an action for each agent. 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. If the set of actions is finite, i.e., $b = |\mathrm{Ac}| < \omega$, we say that $\mathcal{G}$ is b-bounded, or simply bounded. If both the sets of actions and states are finite, we say that $\mathcal{G}$ is finite.

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

A *strategy* is a partial function $\mathsf{f} : \mathrm{Trk} \rightharpoonup \mathrm{Ac}$ that maps each non-trivial track in its domain to an action. For a state $s \in \mathrm{St}$, a strategy f is said *s-total* if it is defined on all tracks starting in s, i.e., $\mathsf{dom}(\mathsf{f}) = \mathrm{Trk}(s)$. $\mathrm{Str} \triangleq \mathrm{Trk} \rightharpoonup \mathrm{Ac}$ (resp., $\mathrm{Str}(s) \triangleq \mathrm{Trk}(s) \to \mathrm{Ac}$) denotes the set of all (resp., s-total) strategies. For all tracks $\rho \in \mathrm{Trk}$, by $(\mathsf{f})_\rho \in \mathrm{Str}$ we denote the *translation* of f along ρ, i.e., the strategy with $\mathsf{dom}((\mathsf{f})_\rho) \triangleq \{\mathsf{lst}(\rho) \cdot \rho' : \rho \cdot \rho' \in \mathsf{dom}(\mathsf{f})\}$ [1] such that $(\mathsf{f})_\rho(\mathsf{lst}(\rho) \cdot \rho') \triangleq \mathsf{f}(\rho \cdot \rho')$, for all $\rho \cdot \rho' \in \mathsf{dom}(\mathsf{f})$.

Let Var be a fixed set of *variables*. An *assignment* is a partial function $\chi : \mathrm{Var} \cup \mathrm{Ag} \rightharpoonup \mathrm{Str}$ mapping variables and agents in its domain to a strategy. An assignment χ is *complete* if it is defined on all agents, i.e., $\mathrm{Ag} \subseteq \mathsf{dom}(\chi)$. For a state $s \in \mathrm{St}$, it is said that χ is *s-total* if all strategies $\chi(l)$ are s-total, for $l \in \mathsf{dom}(\chi)$. $\mathrm{Asg} \triangleq \mathrm{Var} \cup \mathrm{Ag} \rightharpoonup \mathrm{Str}$ (resp., $\mathrm{Asg}(s) \triangleq \mathrm{Var} \cup \mathrm{Ag} \rightharpoonup \mathrm{Str}(s)$) denotes the set of all (resp., s-total) assignments. Moreover, $\mathrm{Asg}(\mathrm{X}) \triangleq \mathrm{X} \to \mathrm{Str}$ (resp., $\mathrm{Asg}(\mathrm{X}, s) \triangleq \mathrm{X} \to \mathrm{Str}(s)$) indicates the subset of X-*defined* (resp., s-total) assignments, i.e., (resp., s-total) assignments defined on the set $\mathrm{X} \subseteq \mathrm{Var} \cup \mathrm{Ag}$. For all tracks $\rho \in \mathrm{Trk}$, by $(\chi)_\rho \in \mathrm{Asg}(\mathsf{lst}(\rho))$ we denote the *translation* of χ along ρ, i.e., the $\mathsf{lst}(\rho)$-total assignment with $\mathsf{dom}((\chi)_\rho) \triangleq \mathsf{dom}(\chi)$, such that $(\chi)_\rho(l) \triangleq (\chi(l))_\rho$, for all $l \in \mathsf{dom}(\chi)$. For all elements $l \in \mathrm{Var} \cup \mathrm{Ag}$, by $\chi[l \mapsto \mathsf{f}] \in \mathrm{Asg}$ 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 χ otherwise, i.e., $\chi[l \mapsto \mathsf{f}](l) \triangleq \mathsf{f}$ and $\chi[l \mapsto \mathsf{f}](l') \triangleq \chi(l')$, for all $l' \in \mathsf{dom}(\chi) \setminus \{l\}$.

A path $\pi \in \mathrm{Pth}(s)$ starting at a state $s \in \mathrm{St}$ is a *play* w.r.t. a complete s-total assignment $\chi \in \mathrm{Asg}(s)$ (*(χ, s)-play*, for short) if, for all $i \in \mathbb{N}$, it holds that $(\pi)_{i+1} = \tau((\pi)_i, \mathsf{d})$, where $\mathsf{d}(a) \triangleq \chi(a)((\pi)_{\leq i})$, for each $a \in \mathrm{Ag}$. The partial function play :

[1] By $\mathsf{lst}(\rho) \triangleq (\rho)_{|\rho|-1}$ we denote the last state of ρ.

$\mathrm{Asg} \times \mathrm{St} \rightharpoonup \mathrm{Pth}$, 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 (χ, s) in its domain.

For a 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 $\pi = \mathsf{play}(\chi, s)$.

From now on, we use CGS names with subscript to extract the components from their tuple-structures. For example, $s_{0\mathcal{G}} = \mathsf{s}_0$ is the starting state of the CGS $\mathcal{G}$.

3 One-Goal Strategy Logic

In this section, we introduce syntax and semantics of One-Goal Strategy Logic (SL[1G], for short), as a syntactic fragment of SL, which we also report here for technical reasons. For more about SL[1G], see [16].

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

Definition 1 (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* $a \in \mathrm{Ag}$*:*

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

By $\mathsf{sub}(\varphi)$ we denote the set of all *subformulas* of the SL formula φ. By $\mathsf{free}(\varphi)$ we denote the set of *free agents/variables* of φ defined as the subset of $\mathrm{Ag} \cup \mathrm{Var}$ containing *(i)* all agents a for which there is no binding (a, x) before the occurrence of a temporal operator and *(ii)* all variables x for which there is a binding (a, x) but no quantification $\langle\!\langle x \rangle\!\rangle$ or $[\![x]\!]$. A formula φ without free agents (resp., variables), i.e., with $\mathsf{free}(\varphi) \cap \mathrm{Ag} = \emptyset$ (resp., $\mathsf{free}(\varphi) \cap \mathrm{Var} = \emptyset$), is named *agent-closed* (resp., *variable-closed*). If φ is both agent- and variable-closed, it is named *sentence*. By $\mathsf{snt}(\varphi)$ we denote the set of all sentences that are subformulas of φ.

SL Semantics As for ATL*, we define the semantics of SL w.r.t. concurrent game structures. For a CGS $\mathcal{G}$, a state 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 under the assignment χ. The semantics of SL formulas involving p, $\neg$, $\wedge$, and $\vee$ is defined as usual in LTL and we omit it here (see [16], for the full definition). The semantics of the remaining part, which involves quantifications, bindings, and temporal operators follows.

Definition 2 (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 relation* $\mathcal{G}, \chi, s \models \varphi$ *is inductively defined as follows.*

1. $\mathcal{G}, \chi, s \models \langle\!\langle x \rangle\!\rangle \varphi$ *iff there is an* s*-total strategy* $\mathsf{f} \in \mathrm{Str}(s)$ *such that* $\mathcal{G}, \chi[x \mapsto \mathsf{f}], s \models \varphi$*;*
2. $\mathcal{G}, \chi, s \models [\![x]\!] \varphi$ *iff for all* s*-total strategies* $\mathsf{f} \in \mathrm{Str}(s)$ *it holds that* $\mathcal{G}, \chi[x \mapsto \mathsf{f}], s \models \varphi$*.*

Moreover, if $\mathsf{free}(\varphi) \cup \{x\} \subseteq \mathsf{dom}(\chi) \cup \{a\}$ *for an agent* $a \in \mathrm{Ag}$*, it holds that:*

3. $\mathcal{G}, \chi, s \models (a, x)\varphi$ *iff* $\mathcal{G}, \chi[a \mapsto \chi(x)], s \models \varphi$.

Finally, if χ *is also complete, it holds that:*

4. $\mathcal{G}, \chi, s \models \mathsf{X}\,\varphi$ *if* $\mathcal{G}, (\chi, s)^1 \models \varphi$*;*
5. $\mathcal{G}, \chi, s \models \varphi_1 \mathsf{U}\,\varphi_2$ *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* $\mathcal{G}, (\chi, s)^j \models \varphi_1$*;*
6. $\mathcal{G}, \chi, s \models \varphi_1 \mathsf{R}\,\varphi_2$ *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 1 and 2, 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 3, 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 the LTL semantics is simply embedded into the SL one.

A CGS $\mathcal{G}$ is a *model* of an SL sentence φ, denoted by $\mathcal{G} \models \varphi$, iff $\mathcal{G}, \emptyset, s_0 \models \varphi$, where $\emptyset$ is the empty assignment. Moreover, φ is *satisfiable* iff there is a model for it. Given two CGSs $\mathcal{G}_1$, $\mathcal{G}_2$ and a sentence φ, we say that φ is *invariant* under $\mathcal{G}_1$ and $\mathcal{G}_2$ iff it holds that: $\mathcal{G}_1 \models \varphi$ iff $\mathcal{G}_2 \models \varphi$. Finally, given two SL formulas φ_1 and φ_2 with $\mathsf{free}(\varphi_1) = \mathsf{free}(\varphi_2)$, we say that φ_1 *implies* φ_2, in symbols $\varphi_1 \Rightarrow \varphi_2$, if, for all CGSs $\mathcal{G}$, states $s \in \mathrm{St}$, 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 $\varphi_1 \Rightarrow \varphi_2$ and $\varphi_2 \Rightarrow \varphi_1$.

As an example, consider the SL sentence $\varphi = \langle\!\langle \mathsf{x} \rangle\!\rangle [\![\mathsf{y}]\!] \langle\!\langle \mathsf{z} \rangle\!\rangle ((\alpha, \mathsf{x})(\beta, \mathsf{y})(\mathsf{X}\,\mathsf{p}) \wedge (\alpha, \mathsf{y})(\beta, \mathsf{z})(\mathsf{X}\,\mathsf{q}))$. Note that both agents α and β use the strategy associated with y to achieve simultaneously the LTL goals $\mathsf{X}\,\mathsf{p}$ and $\mathsf{X}\,\mathsf{q}$, respectively. A model for φ is the CGS $\mathcal{G} \triangleq \langle \{\mathsf{p}, \mathsf{q}\}, \{\alpha, \beta\}, \{0, 1\}, \{\mathsf{s}_0, \mathsf{s}_1, \mathsf{s}_2, \mathsf{s}_3\}, \lambda, \tau, \mathsf{s}_0 \rangle$, 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 go to s_0. See the representation of $\mathcal{G}$ depicted in Figure 1, in which 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. Clearly, $\mathcal{G} \models \varphi$ by letting, on s_0, the variables x to chose action 0 (the formula $(\alpha, \mathsf{x})(\beta, \mathsf{y})(\mathsf{X}\,\mathsf{p})$ 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 formula $(\alpha, \mathsf{y})(\beta, \mathsf{z})(\mathsf{X}\,\mathsf{q})$ is satisfied, since one can move from s_0 to either s_2 or s_3, both labeled with q).

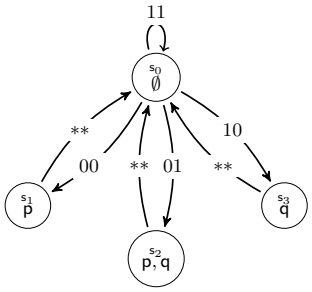

Figure 1. A CGS $\mathcal{G}$.

SL[1G] Syntax To formalize the syntactic fragment SL[1G] of SL, we need first to define the concepts of *quantification* and *binding prefixes*.

Definition 3 (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* $\wp$*. A* binding prefix *over a set* $\mathrm{V} \subseteq \mathrm{Var}$ *of variables 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 $\flat$. *Finally,* $\mathrm{Qnt(V)} \subseteq \{\langle\langle x\rangle\rangle, [\![x]\!] : x \in \mathrm{V}\}^{|\mathrm{V}|}$ *and* $\mathrm{Bnd(V)} \subseteq \{(a, x) : a \in \mathrm{Ag} \wedge x \in \mathrm{V}\}^{|\mathrm{Ag}|}$ *denote, respectively, the sets of all quantification and binding prefixes over variables in* V.

We can now define the syntactic fragment we want to analyze. The idea is to force each group of agent bindings, represented by a binding prefix, to be coupled with a quantification prefix.

Definition 4 (SL[1G] Syntax). SL[1G] *formulas are built inductively from the sets of atomic propositions* AP*, quantification prefixes* $\mathrm{Qnt(V)}$*, for* $\mathrm{V} \subseteq \mathrm{Var}$*, and binding prefixes* $\mathrm{Bnd(Var)}$*, by using the following grammar, with* $p \in \mathrm{AP}$*,* $\wp \in \cup_{\mathrm{V}\subseteq\mathrm{Var}}\mathrm{Qnt(V)}$*, and* $\flat \in \mathrm{Bnd(Var)}$*:*

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

with $\wp \in \mathrm{Qnt}(\mathsf{free}(\flat\varphi))$*, in the formation rule* $\wp\flat\varphi$.

In the following, for a *goal* we mean an SL agent-closed formula of the kind $\flat\psi$, where ψ is variable-closed and $\flat \in \mathrm{Bnd}(\mathsf{free}(\psi))$. Note that, since $\flat\varphi$ is a goal, it is agent-closed, so, $\mathsf{free}(\flat\varphi) \subseteq \mathrm{Var}$. Moreover, an SL[1G] sentence φ is *principal* if it is of the form $\varphi = \wp\flat\psi$, where $\flat\psi$ is a goal and $\wp \in \mathrm{Qnt}(\mathsf{free}(\flat\psi))$. By $\mathsf{psnt}(\varphi) \subseteq \mathsf{snt}(\varphi)$ we denote the set of *principal subsentences* of the SL[1G] formula φ.

As an example, let $\varphi_1 = \wp\flat_1\psi_1$ and $\varphi_2 = \wp(\flat_1\psi_1 \wedge \flat_2\psi_2)$, where $\wp = [\![\mathsf{x}]\!]\langle\langle\mathsf{y}\rangle\rangle[\![\mathsf{z}]\!]$, $\flat_1 = (\alpha, \mathsf{x})(\beta, \mathsf{y})(\gamma, \mathsf{z})$, $\flat_2 = (\alpha, \mathsf{y})(\beta, \mathsf{z})(\gamma, \mathsf{y})$, $\psi_1 = \mathsf{X}\,\mathsf{p}$, and $\psi_2 = \mathsf{X}\,\mathsf{q}$. Then, it is evident that $\varphi_1 \in$ SL[1G] but $\varphi_2 \notin$ SL[1G], since the quantification prefix $\wp$ of the latter does not have in its scope a unique goal.

It is fundamental to observe that the formula φ_1 of the above example cannot be expressed in ATL*, as proved in [16] and reported in the following theorem, since its 2-quantifier alternation cannot be encompassed in the 1-alternation ATL* modalities. On the contrary, each ATL* formula of the type $\langle\langle \mathrm{A}\rangle\rangle\psi$, where $\mathrm{A} = \{\alpha_1, \ldots, \alpha_n\} \subseteq \mathrm{Ag} = \{\alpha_1, \ldots, \alpha_n, \beta_1, \ldots, \beta_m\}$ can be expressed in SL[1G] as follows: $\langle\langle \mathsf{x}_1\rangle\rangle \cdots \langle\langle x_n\rangle\rangle[\![\mathsf{y}_1]\!] \cdots [\![\mathsf{y}_m]\!](\alpha_1, \mathsf{x}_1) \cdots (\alpha_n, \mathsf{x}_n)(\beta_1, \mathsf{y}_1) \cdots (\beta_m, \mathsf{y}_m)\psi$.

Theorem 1. SL[1G] *is strictly more expressive than* ATL*.

We now give two examples in which we show the importance of the ability to write specifications with alternation of quantifiers greater than 1 along with strategy sharing.

Example 1 (Escape from Alcatraz[2]*).* Consider the situation in which an Alcatraz prisoner tries to escape from jail by helicopter with the help of an accomplice. Due to his panoramic point of view, assume that the accomplice has the full visibility on the behaviors of guards, while the prisoner does not have the same ability. Therefore, the latter has to put in practice an escape strategy that, independently of guards moves, can be supported by his accomplice to escape. We can formalize such an intricate situation by means of an SL[1G] sentence with alternation 2, where the prisoner has to choose a uniform strategy w.r.t. those chosen by the guards, as follows. First, let $\mathcal{G}_A$ be a CGS modeling the possible situations in which the agents "p" prisoner, "g" guards, and "a" accomplice can reside, together with all related possible moves. Then, verify the existence of an escape strategy by checking $\mathcal{G}_A \models \langle\langle\mathsf{x}\rangle\rangle[\![\mathsf{y}]\!]\langle\langle\mathsf{z}\rangle\rangle(\mathsf{p}, \mathsf{x})(\mathsf{g}, \mathsf{y})(\mathsf{a}, \mathsf{z})(\mathsf{F}\ \mathsf{free}_{\mathsf{P}})$.

[2] We thank Luigi Sauro for having pointed out this example.

Example 2 (Stealing-Strategy in Hex). Hex is a two-player game, red vs 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, i.e., 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[1G] as follows. First model Hex with a CGS $\mathcal{G}_H$ whose states represent a possible possible configurations reached during a play between "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, i.e., to be the first to connect the related red edges ($\mathsf{F}\ \mathsf{cnc}_\mathsf{r}$).

4 Strategy Quantifications

We now define the concept of *dependence map*. The key idea is that every quantification prefix of an SL formula can be represented by a suitable choice of a dependence map over strategies. Such a result is at the base of the definition of the *elementariness* property and allows us to prove that SL[1G] is elementarily satisfiable, i.e., we can simplify a reasoning about strategies by reducing it to a set of local reasonings about actions [16].

Dependence map First, we introduce some notation regarding quantification prefixes. Let $\wp \in \mathrm{Qnt}(\mathrm{V})$ be a quantification prefix over a set $\mathrm{V}(\wp) \triangleq \mathrm{V} \subseteq \mathrm{Var}$ of variables. By $\langle\langle \wp \rangle\rangle \triangleq \{x \in \mathrm{V} : \exists i \in [0, |\wp|[.(\wp)_i = \langle\langle x \rangle\rangle\}$ and $[\![\wp]\!] \triangleq \mathrm{V} \setminus \langle\langle \wp \rangle\rangle$ we denote, respectively, the sets of *existential* and *universal variables* quantified in $\wp$. For two variables $x, y \in \mathrm{V}$, we say that x *precedes* y in $\wp$, in symbols $x <_\wp y$, if x occurs before y in $\wp$. Moreover, by $\mathrm{Dep}(\wp) \triangleq \{(x,y) \in \mathrm{V} \times \mathrm{V} : x \in [\![\wp]\!], y \in \langle\langle \wp \rangle\rangle \wedge x <_\wp y\}$ we denote the set of *dependence pairs*, i.e., a dependence relation, on which we derive the parameterized version $\mathrm{Dep}(\wp, y) \triangleq \{x \in \mathrm{V} : (x,y) \in \mathrm{Dep}(\wp)\}$ containing all variables from which y depends. Also, we use $\overline{\wp} \in \mathrm{Qnt}(\mathrm{V})$ to indicate the quantification derived from $\wp$ by *dualizing* each quantifier contained in it, i.e., for all $i \in [0, |\wp|[$, it holds that $(\overline{\wp})_i = \langle\langle x \rangle\rangle$ iff $(\wp)_i = [\![x]\!]$, with $x \in \mathrm{V}$. Clearly, $\langle\langle \overline{\wp} \rangle\rangle = [\![\wp]\!]$ and $[\![\overline{\wp}]\!] = \langle\langle \wp \rangle\rangle$. Finally, we define the notion of *valuation* of variables over a generic set D as 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 $\mathrm{V} \subseteq \mathrm{Var}$.

We now give the semantics for quantification prefixes via the following definition of *dependence map*.

Definition 5 (Dependence Maps). *Let $\wp \in \mathrm{Qnt}(\mathrm{V})$ be a quantification prefix over a set of variables $\mathrm{V} \subseteq \mathrm{Var}$, and D a set. Then, a* dependence map *for $\wp$ over D is a function $\theta : \mathrm{Val}_\mathrm{D}([\![\wp]\!]) \to \mathrm{Val}_\mathrm{D}(\mathrm{V})$ satisfying the following properties:* (i) $\theta(\mathsf{v})_{\restriction [\![\wp]\!]} = \mathsf{v}$, *for all* $\mathsf{v} \in \mathrm{Val}_\mathrm{D}([\![\wp]\!])$; (ii) $\theta(\mathsf{v}_1)(x) = \theta(\mathsf{v}_2)(x)$, *for all* $\mathsf{v}_1, \mathsf{v}_2 \in \mathrm{Val}_\mathrm{D}([\![\wp]\!])$ *and* $x \in \langle\langle \wp \rangle\rangle$ *such that* $\mathsf{v}_{1 \restriction \mathrm{Dep}(\wp,x)} = \mathsf{v}_{2 \restriction \mathrm{Dep}(\wp,x)}$. $\mathrm{DM}_\mathrm{D}(\wp)$ *denotes the set of all dependence maps of $\wp$ on* D.

Intuitively, Item (i) asserts that θ takes the same values of its argument w.r.t. the universal variables in $\wp$ and Item (ii) ensures that the value of θ w.r.t. an existential variable x in $\wp$

does not depend on variables not in $\mathrm{Dep}(\wp, x)$. To get better insight into this definition, a dependence map θ for $\wp$ can be considered as a set of *Skolem functions* that, given a value for each universal variable, return a possible value for all the existential variables in a way that is consistent w.r.t. the order of quantifications in $\wp$.

We now state a fundamental theorem that describes how to eliminate strategy quantifications of an SL formula via a choice of a dependence map over strategies. This procedure, easily proved to be correct by induction on the structure of the formula in [16], can be seen as the equivalent of the *Skolemization* in first order logic [10].

Theorem 2 (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, $\mathcal{G} \models \varphi$ iff there exists a dependence map $\theta \in \mathrm{DM}_{\mathrm{Str}(s_0)}(\wp)$ such that $\mathcal{G}, \theta(\chi), s_0 \models \psi$, for all $\chi \in \mathrm{Asg}([\![\wp]\!], s_0)$.*

The above theorem substantially characterizes SL semantics by means of the concept of dependence map. In particular, it shows that if a formula is satisfiable then it is always possible to find a suitable dependence map returning the existential strategies in response to the universal ones. Such a characterization enables the definition of an alternative semantics of SL, based on the choice of a subset of dependence maps that meet a certain given property. We do this with the aim of identifying semantic fragments of SL having better model properties and easier decision problems. With more details, given a CGS $\mathcal{G}$, one of its states s, and a property $\mathtt{P}$, we say that a sentence $\wp\psi$ is $\mathtt{P}$-satisfiable in $\mathcal{G}$, in symbols $\mathcal{G} \models_{\mathtt{P}} \wp\psi$, if there exists a dependence map θ meeting $\mathtt{P}$ such that, for all assignment $\chi \in \mathrm{Asg}([\![\wp]\!], s)$, it holds that $\mathcal{G}, \theta(\chi), s \models \psi$. An alternative semantics identified by a property $\mathtt{P}$ is even more interesting if there exists a syntactic fragment corresponding to it, i.e., each satisfiable sentence of such a fragment is $\mathtt{P}$-satisfiable and vice versa. In the following, we put in practice this idea in order to show that SL[1G] has the same complexity of ATL* w.r.t. the satisfiability problem.

Elementary quantifications According to the above description, we now introduce a suitable property of dependence maps, called elementariness, together with the related alternative semantics. Then, in Theorem 3, we state that SL[1G] has the elementariness property, i.e., each SL[1G] sentence is satisfiable iff it is elementarily satisfiable. Intuitively, a dependence map $\theta \in \mathrm{DM}_{\mathrm{T}\to\mathrm{D}}(\wp)$ over functions from a set T to a set D is elementary if it can be split into a set of dependence maps over D, one for each element of T, represented by a function $\widetilde{\theta} : \mathrm{T} \to \mathrm{DM}_{\mathrm{D}}(\wp)$. This idea allows us to greatly simplify the reasoning about strategy quantifications, since we can reduce them to a set of quantifications over actions, one for each track in their domains.

Note that sets D and T, as well as U and V used in the following, are generic and in our framework they may refer to actions and strategies (D), tracks (T), and variables (U and V). In particular, observe that functions from T to D represent strategies. We prefer to use abstract name, as the properties we describe hold generally.

To formally develop the above idea, we have first to introduce the generic concept of *adjoint function*. From now on, we denote by $\widehat{\mathsf{g}} : \mathrm{Y} \to (\mathrm{X} \to \mathrm{Z})$ the operation of *flipping* of a generic function $\mathsf{g} : \mathrm{X} \to (\mathrm{Y} \to \mathrm{Z})$, i.e., the transformation of g by swapping the order of its arguments. Such a flipping is well-grounded due to the following chain of isomorphisms: $\mathrm{X} \to (\mathrm{Y} \to \mathrm{Z}) \cong (\mathrm{X} \times \mathrm{Y}) \to \mathrm{Z} \cong (\mathrm{Y} \times \mathrm{X}) \to \mathrm{Z} \cong \mathrm{Y} \to (\mathrm{X} \to \mathrm{Z})$.

Definition 6 (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* $t \in \mathrm{T}$*.*

Intuitively, a function m transforming a map of kind $(\mathrm{T} \to \mathrm{D})^{\mathrm{U}}$ into a new map of kind $(\mathrm{T} \to \mathrm{D})^{\mathrm{V}}$ has an adjoint $\widetilde{\mathsf{m}}$ if such a transformation can be done pointwisely w.r.t. the set T, i.e., we can put out as a common domain the set T and then transform a map of kind D^{U} in a map of kind D^{V}. Observe that, if a function has an adjoint, this is unique. Similarly, from an adjoint function it is possible to determine the original function unambiguously. Thus, it is established a one-to-one correspondence between functions admitting an adjoint and the adjoint itself.

The formal meaning of the elementariness of a dependence map over generic functions follows.

Definition 7 (Elementary Dependence Maps). *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* $\theta \in \mathrm{DM}_{\mathrm{T} \to \mathrm{D}}(\wp)$ *a dependence map for* $\wp$ *over* $\mathrm{T} \to \mathrm{D}$*. Then,* θ *is* elementary *if it admits an adjoint function.* $\mathrm{EDM}_{\mathrm{T} \to \mathrm{D}}(\wp)$ *denotes the set of all elementary dependence maps for* $\wp$ *over* $\mathrm{T} \to \mathrm{D}$*.*

As mentioned above, we now introduce an important variant of SL[1G] semantics based on the property of elementariness of dependence maps over strategies. We refer to the related satisfiability concept as *elementary satisfiability*, in symbols $\models_{\mathrm{E}}$.

The new semantics of SL[1G] formulas involving atomic propositions, Boolean connectives, temporal operators, and agent bindings is defined as for the classic one, where the modeling relation $\models$ is substituted with $\models_{\mathrm{E}}$, and we omit to report it here. In the following definition, we only describe the part concerning the quantification prefixes. Observe that by $\zeta_{\flat} : \mathrm{Ag} \to \mathrm{Var}$, for $\flat \in \mathrm{Bnd}(\mathrm{Var})$, we denote the function associating to each agent the variable of its binding in $\flat$.

Definition 8 (SL[1G] Elementary Semantics). *Let* $\mathcal{G}$ *be a* CGS*,* $s \in \mathrm{St}$ *one of its states, and* $\wp \flat \psi$ *an* SL[1G] *principal sentence. Then* $\mathcal{G}, \varnothing, s \models_{\mathrm{E}} \wp \flat \psi$ *iff there is an elementary dependence map* $\theta \in \mathrm{EDM}_{\mathrm{Str}(s)}(\wp)$ *for* $\wp$ *over* $\mathrm{Str}(s)$ *such that* $\mathcal{G}, \theta(\chi) \circ \zeta_{\flat}, s \models_{\mathrm{E}} \psi$*, for all* $\chi \in \mathrm{Asg}(\llbracket \wp \rrbracket, s)$*.*

It is immediate to see a strong similarity between the statement of Theorem 2 of SL strategy quantification and the previous definition. The only crucial difference resides in the choice of the kind of dependence map. Moreover, observe that, differently from the classic semantics, the quantifications in a prefix are not treated individually but as an atomic block. This is due to the necessity of having a strict correlation between the point-wise structure of the quantified strategies.

Finally, we state the following fundamental theorem which is a key step in the proof of the bounded model property and decidability of the satisfiability for SL[1G], whose correctness has been proved in [16]. The idea behind the proof of the elementariness property resides in the strong similarity between the statement of Theorem 2 of SL strategy quantification and the definition of the winning condition in a classic turn-based two-player game. Indeed, on one hand, we say that a sentence is satisfiable iff "there exists a dependence map such that, for all all assignments, it holds that ...". On the other hand, we say that the first player wins a game iff "there exists a strategy for him such

that, for all strategies of the other player, it holds that ...". The gap between these two formulations is solved in SL[1G] by using the concept of elementary quantification. So, we build a two-player turn-based game in which the two players are viewed one as a dependence map and the other as a valuation over universal quantified variables, both over actions, such that the formula is satisfied iff the first player wins the game. This construction is a deep technical evolution of the proof method used for the dualization of alternating automata on infinite objects [20]. Precisely, it uses Martin's Determinacy Theorem [15] on the auxiliary turn-based game to prove that, if there is no dependence map of a given prefix that satisfies the given property, there is a dependence map of the dual prefix satisfying its negation.

Theorem 3 (SL[1G] Elementariness). *Let $\mathcal{G}$ be a* CGS *and φ an* SL[1G] *sentence. Then, $\mathcal{G} \models \varphi$ iff $\mathcal{G} \models_{\mathrm{E}} \varphi$.*

In order to understand what elementariness means from a syntactic point of view, note that in SL[1G] it holds that $\wp\flat\mathsf{X}\,\psi \equiv \wp\flat\mathsf{X}\,\wp\flat\psi$, i.e., we can requantify the strategies to satisfy the inner subformula ψ. This equivalence is a generalization of what is well know to hold for CTL*: $\mathsf{EX}\,\psi \equiv \mathsf{EX}\,\mathsf{E}\psi$. Moreover, note that, as reported in [16], elementariness does not hold for more expressive fragments of SL, such as SL[BG].

5 Model Properties

We now investigate basic model properties of SL[1G] that turn out to be important on their own and useful to prove the decidability of the satisfiability problem.

First, recall that the satisfiability problem for branching-time logics can be solved via tree automata, once a kind of bounded tree-model property holds. Indeed, by using it, one can build an automaton accepting all models of formulas, or their encoding. So, we first introduce the concepts of *concurrent game tree*, *decision tree*, and *decision-unwinding* and then show that SL[1G] is *invariant under decision-unwinding*, which directly implies that it satisfies an *unbounded tree-model property*. Finally, by using a sharp technique that is precisely described in [17], we further prove that the above property is actually a *bounded tree-model property*.

Tree-model property We now introduce two particular kinds of CGS whose structure is a directed tree. As already explained, we do this since the decidability procedure we give in the last section of the paper is based on alternating tree automata.

Definition 9 (Concurrent Game Trees). *A* concurrent game tree *(*CGT*, for short) is a* CGS $\mathcal{T} \triangleq \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}, \mathrm{St}, \lambda, \tau, \varepsilon \rangle$*, where* (i) $\mathrm{St} \subseteq \Delta^*$ *is a Δ-tree for a given set Δ of directions and* (ii) *if $t \cdot e \in \mathrm{St}$ then there is a decision $\mathsf{d} \in \mathrm{Dc}$ such that $\tau(t, \mathsf{d}) = t \cdot e$, for all $t \in \mathrm{St}$ and $e \in \Delta$. Furthermore, $\mathcal{T}$ is a* decision tree *(*DT*, for short) if* (i) $\mathrm{St} = \mathrm{Dc}^*$ *and* (ii) *if $t \cdot \mathsf{d} \in \mathrm{St}$ then $\tau(t, \mathsf{d}) = t \cdot \mathsf{d}$, for all $t \in \mathrm{St}$ and $\mathsf{d} \in \mathrm{Dc}$.*

Intuitively, CGTs are CGSs with a tree-shaped transition relation and DTs have, in addition, states uniquely determining the history of computation leading to them.

At this point, we can define a generalization for CGSs of the classic concept of *unwinding* of labeled transition systems, namely decision-unwinding. Note that, in

general and differently from ATL*, SL is not invariant under decision-unwinding, as we show later. On the contrary, SL[1G] satisfies such an invariance property. This fact allows us to show that this logic has the unbounded tree-model property.

Definition 10 (Decision-Unwinding). *Let $\mathcal{G}$ be a* CGS*. Then, the* decision-unwinding *of $\mathcal{G}$ is the* DT $\mathcal{G}_{DU} \triangleq \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}_{\mathcal{G}}, {\mathrm{Dc}_{\mathcal{G}}}^*, \lambda, \tau, \varepsilon \rangle$ *for which there is a surjective function* $\mathsf{unw} : {\mathrm{Dc}_{\mathcal{G}}}^* \to \mathrm{St}_{\mathcal{G}}$ *such that* (i) $\mathsf{unw}(\varepsilon) = s_{0\mathcal{G}}$*,* (ii) $\mathsf{unw}(\tau(t, \mathsf{d})) = \tau_{\mathcal{G}}(\mathsf{unw}(t), \mathsf{d})$*, and* (iii) $\lambda(t) = \lambda_{\mathcal{G}}(\mathsf{unw}(t))$*, for all* $t \in {\mathrm{Dc}_{\mathcal{G}}}^*$ *and* $\mathsf{d} \in \mathrm{Dc}_{\mathcal{G}}$*.*

Note that each CGS $\mathcal{G}$ has a unique associated decision-unwinding $\mathcal{G}_{DU}$.

We say that a sentence φ has the *decision-tree model property* if, for each CGS $\mathcal{G}$, it holds that $\mathcal{G} \models \varphi$ iff $\mathcal{G}_{DU} \models \varphi$. By using a standard proof by induction on the structure of SL[1G] formulas, we can show that this logic is invariant under decision-unwinding, i.e., each SL[1G] sentence has decision-tree model property, and, consequently, that it satisfies the unbounded tree-model property. For the case of the combined quantification and binding prefixes $\wp\flat\psi$, we can use a technique that allows to build, given an elementary dependence map θ satisfying the formula on a CGS $\mathcal{G}$, an elementary dependence map θ' satisfying the same formula over the DT $\mathcal{G}_{DU}$, and vice versa. This construction is based on a step-by-step transformation of the adjoint of a dependence maps into another, which is done for each track of the original model. This means that we do not actually transform the strategy quantifications but the equivalent infinite set of action quantifications.

Theorem 4 (SL[1G] Positive Model Properties). *For* SL[1G] *it holds that:* (i) *it is invariant under decision-unwinding and* (ii) *it has the decision-tree model property.*

Although this result is a generalization of that proved to hold for ATL*, it actually represents an important demarcation line between SL[1G] and SL. Indeed, as we show in the following theorem, SL does not satisfy neither the tree-model property nor, consequently, the invariance under decision-unwinding.

Theorem 5 (SL Negative Model Properties). *For* SL *it holds that:* (i) *it does not have the decision-tree model property and* (ii) *it is not invariant under decision-unwinding.*

Bounded tree-model property We now have all tools we need to prove the bounded tree-model property for SL[1G], which we recall SL does not satisfy [18]. Actually, we prove here a stronger property, which we name *bounded disjoint satisfiability*.

To this aim, we first introduce the new concept, called *disjoint satisfiability*, regarding the satisfiability of different instances of the same subsentence of the original specification, which intuitively states that these instances can be checked on disjoint subtrees of the tree model. With more detail, this property asserts that, if two instances use part of the same subtree, they are forced to use the same dependence map as well. This intrinsic characteristic of SL[1G] is fundamental to build a unique automaton that checks the truth of all subsentences, by simply merging their respective automata, without using a projection operation that eliminates their proper alphabets, which otherwise can be in conflict. In this way, we can avoid an exponential blow-up.

In the following theorem, we finally describe the crucial step behind our automata-theoretic decidability procedure for SL[1G]. At an high-level, the proof proceeds as follows. We start from the satisfiability of the specification φ over a DT $\mathcal{T}$, whose

existence is ensured by Item *(ii)* of Theorem 4 of SL[1G] positive model properties. Then, by means of Theorem 3 on the SL[1G] elementariness, we construct the adjoint functions of the dependence maps used to verify the satisfiability of the sentences on $\mathcal{T}$. Finally, by using a fundamental and very technical property of dependence maps, called *overlapping* [17], we transform the dependence maps over actions, contained in the ranges of the adjoint functions, in a bounded version, which preserves the satisfiability of the sentences on a bounded pruning $\mathcal{T}'$ of $\mathcal{T}$.

Theorem 6 (SL[1G] Bounded Tree-Model Property). *Let φ be an* SL[1G] *satisfiable sentence. Then, there exists a bounded* CGT $\mathcal{T}$ *such that $\mathcal{T} \models \varphi$. Moreover, for all $\phi \in \mathsf{psnt}(\varphi)$, it holds that $\mathcal{T}$ satisfies ϕ disjointly over the set $\{s \in \mathrm{St} : \mathcal{T}, \emptyset, s \models \phi\}$.*

6 Satisfiability Procedure

We finally solve the satisfiability problem for SL[1G] and show that it is 2EXPTIME-COMPLETE, as for ATL*. The algorithmic procedures is based on an automata-theoretic approach, which reduces the decision problem for the logic to the emptiness problem of a suitable universal Co-Büchi tree automaton (UCT, for short) [9]. From an high-level point of view, the automaton construction seems similar to what was proposed in literature for CTL* [13] and ATL* [26]. However, our technique is completely new, since it is based on the novel notions of elementariness and disjoint satisfiability.

Principal sentences To proceed with the satisfiability procedure, we have to introduce a concept of encoding for an assignment and the labeling of a DT.

Definition 11 (Assignment-Labeling Encoding). *Let $\mathcal{T}$ be a* DT*, $t \in \mathrm{St}_{\mathcal{T}}$ one of its states, and $\chi \in \mathrm{Asg}_{\mathcal{T}}(\mathrm{V}, t)$ an assignment defined on the set $\mathrm{V} \subseteq \mathrm{Var}$. A $(\mathrm{Val}_{\mathrm{Ac}_{\mathcal{T}}}(\mathrm{V}) \times 2^{\mathrm{AP}})$-labeled $\mathrm{Dc}_{\mathcal{T}}$-tree $\mathcal{T}' \triangleq \langle \mathrm{St}_{\mathcal{T}}, \mathsf{u} \rangle$ is an* assignment-labeling encoding *for χ on $\mathcal{T}$ if $\mathsf{u}(\mathsf{lst}((\rho)_{\geq 1})) = (\widehat{\chi}(\rho), \lambda_{\mathcal{T}}(\mathsf{lst}(\rho)))$, for all $\rho \in \mathrm{Trk}_{\mathcal{T}}(t)$.*

Observe that there is a unique assignment-labeling encoding for each assignment over a given DT.

Now, we prove the existence of a UCT $\mathcal{U}^{\mathrm{Ac}}_{\flat\psi}$ for each SL[1G] goal $\flat\psi$ having no principal subsentences. $\mathcal{U}^{\mathrm{Ac}}_{\flat\psi}$ recognizes all the assignment-labeling encodings $\mathcal{T}'$ of an a priori given assignment χ over a generic DT $\mathcal{T}$, once the goal is satisfied on $\mathcal{T}$ under χ. Intuitively, we start with a UCW, recognizing all infinite words on the alphabet 2^{AP} that satisfy the LTL formula ψ, obtained by a simple variation of the Vardi-Wolper construction [28]. Then, we run it on the encoding tree $\mathcal{T}'$ by following the directions imposed by the assignment in its labeling.

Lemma 1 (SL[1G] Goal Automaton). *Let $\flat\psi$ an* SL[1G] *goal without principal subsentences and* Ac *a finite set of actions. Then, there exists an* UCT $\mathcal{U}^{\mathrm{Ac}}_{\flat\psi} \triangleq \langle \mathrm{Val}_{\mathrm{Ac}}(\mathsf{free}(\flat\psi)) \times 2^{\mathrm{AP}}, \mathrm{Dc}, \mathrm{Q}_{\flat\psi}, \delta_{\flat\psi}, q_{0\flat\psi}, \aleph_{\flat\psi} \rangle$ *such that, for all* DTs *$\mathcal{T}$ with $\mathrm{Ac}_{\mathcal{T}} = \mathrm{Ac}$, states $t \in \mathrm{St}_{\mathcal{T}}$, and assignments $\chi \in \mathrm{Asg}_{\mathcal{T}}(\mathsf{free}(\flat\psi), t)$, it holds that $\mathcal{T}, \chi, t \models \flat\psi$ iff $\mathcal{T}' \in \mathrm{L}(\mathcal{U}^{\mathrm{Ac}}_{\flat\psi})$, where $\mathcal{T}'$ is the assignment-labeling encoding for χ on $\mathcal{T}$.*

We now introduce a new concept of encoding regarding the elementary dependence maps over strategies.

Definition 12 (Elementary Dependence-Labeling Encoding). *Let $\mathcal{T}$ be a* DT*, $t \in \mathrm{St}_{\mathcal{T}}$ one of its states, and $\theta \in \mathrm{EDM}_{\mathrm{Str}_{\mathcal{T}}(t)}(\wp)$ an elementary dependence map over strategies for a quantification prefix $\wp \in \mathrm{Qnt}(\mathrm{V})$ over the set $\mathrm{V} \subseteq \mathrm{Var}$. A $(\mathrm{DM}_{\mathrm{Ac}_{\mathcal{T}}}(\wp) \times 2^{\mathrm{AP}})$-labeled Δ-tree $\mathcal{T}' \triangleq \langle \mathrm{St}_{\mathcal{T}}, \mathsf{u} \rangle$ is an* elementary dependence-labeling encoding *for θ on $\mathcal{T}$ if $\mathsf{u}(\mathsf{lst}((\rho)_{\geq 1})) = (\widetilde{\theta}(\rho), \lambda_{\mathcal{T}}(\mathsf{lst}(\rho)))$, for all $\rho \in \mathrm{Trk}_{\mathcal{T}}(t)$.*

Observe that also in this case there exists a unique elementary dependence-model encoding for each elementary dependence map over strategies.

Finally, 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 modeled by the choice of an action dependence map. Intuitively, we guess in the labeling what is the right part of the dependence map over strategies for each node of the tree and then verify that, for all assignments of universal variables, the corresponding complete assignment satisfies the inner formula.

Lemma 2 (SL[1G] Sentence Automaton). *Let $\wp \flat \psi$ be an* SL[1G] *principal sentence without principal subsentences and* Ac *a finite set of actions. Then, there exists an* UCT $\mathcal{U}^{\mathrm{Ac}}_{\wp \flat \psi} \triangleq \langle \mathrm{DM}_{\mathrm{Ac}}(\wp) \times 2^{\mathrm{AP}}, \mathrm{Dc}, \mathrm{Q}_{\wp \flat \psi}, \delta_{\wp \flat \psi}, q_{0 \wp \flat \psi}, \aleph_{\wp \flat \psi} \rangle$ *such that, for all* DT*s $\mathcal{T}$ with $\mathrm{Ac}_{\mathcal{T}} = \mathrm{Ac}$, states $t \in \mathrm{St}_{\mathcal{T}}$, and elementary dependence maps over strategies $\theta \in \mathrm{EDM}_{\mathrm{Str}_{\mathcal{T}}(t)}(\wp)$, it holds that $\mathcal{T}, \theta(\chi), t \models_{\mathrm{E}} \flat \psi$, for all $\chi \in \mathrm{Asg}_{\mathcal{T}}([\![\wp]\!], t)$, iff $\mathcal{T}' \in \mathrm{L}(\mathcal{U}^{\mathrm{Ac}}_{\wp \flat \psi})$, where $\mathcal{T}'$ is the elementary dependence-labeling encoding for θ on $\mathcal{T}$.*

Full sentences By summing up all previous results, we are now able to solve the satisfiability problem for the full SL[1G] fragment.

To construct the automaton for a given SL[1G] sentence φ, we first consider all UCT $\mathcal{U}^{\mathrm{Ac}}_{\phi}$, for an assigned bounded set Ac, previously described for the principal sentences $\phi \in \mathsf{psnt}(\varphi)$, in which the inner subsentences are considered as atomic propositions. Then, thanks to the disjoint satisfiability property, we can merge them into a unique UCT $\mathcal{U}_{\varphi}$ that supplies the dependence map labeling of internal components $\mathcal{U}^{\mathrm{Ac}}_{\phi}$, by using the two functions head and body contained into its labeling. Moreover, observe that the final automaton runs on a b-bounded decision tree, where b is obtained from Theorem 6 on the bounded-tree model property.

Theorem 7 (SL[1G] Automaton). *Let φ be an* SL[1G] *sentence. Then, there exists an* UCT $\mathcal{U}_{\varphi}$ *such that φ is satisfiable iff $\mathrm{L}(\mathcal{U}_{\varphi}) \neq \emptyset$.*

Finally, by a simple calculation of the size of $\mathcal{U}_{\varphi}$ and the complexity of the related emptiness problem, we state in the next theorem the precise computational complexity of the satisfiability problem for SL[1G].

Theorem 8 (SL[1G] Satisfiability). *The satisfiability problem for* SL[1G] *is* 2EXPTIME-COMPLETE.

References

[1] M.H. Albert, R.J. Nowakowski, and D. Wolfe. *Lessons in Play: An Introduction to Combinatorial Game Theory*. AK Peters, 2007.

[2] R. Alur, T.A. Henzinger, and O. Kupferman. Alternating-Time Temporal Logic. *JACM*, 49(5):672–713, 2002.

[3] K. Chatterjee, T.A. Henzinger, and N. Piterman. Strategy Logic. *IC*, 208(6):677–693, 2010.
[4] E.M. Clarke, O. Grumberg, and D.A. Peled. *Model Checking.* MIT Press, 2002.
[5] A. Da Costa, F. Laroussinie, and N. Markey. ATL with Strategy Contexts: Expressiveness and Model Checking. In *FSTTCS'10*, LIPIcs 8, pages 120–132, 2010.
[6] E.A. Emerson and J.Y. Halpern. "Sometimes" and "Not Never" Revisited: On Branching Versus Linear Time. *JACM*, 33(1):151–178, 1986.
[7] B. Finkbeiner and S. Schewe. Coordination Logic. In *CSL'10*, LNCS 6247, pages 305–319. Springer, 2010.
[8] D. Fisman, O. Kupferman, and Y. Lustig. Rational Synthesis. In *TACAS'10*, LNCS 6015, pages 190–204. Springer, 2010.
[9] E. Grädel, W. Thomas, and T. Wilke. *Automata, Logics, and Infinite Games: A Guide to Current Research.* LNCS 2500. Springer-Verlag, 2002.
[10] W. Hodges. *Model theory.* Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1993.
[11] W. Jamroga and W. van der Hoek. Agents that Know How to Play. *FI*, 63(2-3):185–219, 2004.
[12] D. Kozen. Results on the Propositional mu-Calculus. *TCS*, 27(3):333–354, 1983.
[13] O. Kupferman, M.Y. Vardi, and P. Wolper. An Automata Theoretic Approach to Branching-Time Model Checking. *JACM*, 47(2):312–360, 2000.
[14] O. Kupferman, M.Y. Vardi, and P. Wolper. Module Checking. *IC*, 164(2):322–344, 2001.
[15] A.D. Martin. Borel Determinacy. *AM*, 102(2):363–371, 1975.
[16] F. Mogavero, A. Murano, G. Perelli, and M.Y. Vardi. Reasoning About Strategies: On the Model-Checking Problem. Technical Report 1112.6275, arXiv, December 2011.
[17] F. Mogavero, A. Murano, G. Perelli, and M.Y. Vardi. A Decidable Fragment of Strategy Logic. Technical Report 1202.1309, arXiv, February 2012.
[18] F. Mogavero, A. Murano, and M.Y. Vardi. Reasoning About Strategies. In *FSTTCS'10*, LIPIcs 8, pages 133–144, 2010.
[19] F. Mogavero, A. Murano, and M.Y. Vardi. Relentful Strategic Reasoning in Alternating-Time Temporal Logic. In *LPAR'10*, LNAI 6355, pages 371–387. Springer, 2010.
[20] D.E. Muller and P.E. Schupp. Alternating Automata on Infinite Trees. *TCS*, 54(2-3):267–276, 1987.
[21] D.E. Muller and P.E. Schupp. Simulating Alternating Tree Automata by Nondeterministic Automata: New Results and New Proofs of Theorems of Rabin, McNaughton, and Safra. *TCS*, 141(1-2):69–107, 1995.
[22] M. Pauly. A Modal Logic for Coalitional Power in Games. *JLC*, 12(1):149–166, 2002.
[23] S. Pinchinat. A Generic Constructive Solution for Concurrent Games with Expressive Constraints on Strategies. In *ATVA'07*, LNCS 4762, pages 253–267. Springer, 2007.
[24] A. Pnueli. The Temporal Logic of Programs. In *FOCS'77*, pages 46–57, 1977.
[25] M.O. Rabin. Decidability of Second-Order Theories and Automata on Infinite Trees. *TAMS*, 141:1–35, 1969.
[26] S. Schewe. ATL* Satisfiability is 2ExpTime-Complete. In *ICALP'08*, LNCS 5126, pages 373–385. Springer, 2008.
[27] M.Y. Vardi. Why is Modal Logic So Robustly Decidable? In *DCFM'96*, pages 149–184. American Mathematical Society, 1996.
[28] M.Y. Vardi and P. Wolper. An Automata-Theoretic Approach to Automatic Program Verification. In *LICS'86*, pages 332–344. IEEE Computer Society, 1986.
[29] M.Y. Vardi and P. Wolper. Automata-Theoretic Techniques for Modal Logics of Programs. *JCSS*, 32(2):183–221, 1986.
[30] F. Wang, C. Huang, and F. Yu. A Temporal Logic for the Interaction of Strategies. In *CONCUR'11*, LNCS 6901, pages 466–481. Springer, 2011.