On the Boundary of Behavioral Strategies[*]

Fabio Mogavero, Aniello Murano, and Luigi Sauro
Università degli Studi di Napoli Federico II

***Abstract*—In the setting of multi-agent games, considerable effort has been devoted to the definition of modal logics for strategic reasoning. In this area, a recent contribution is given by the introduction of *Strategy Logic* (SL, for short) by Mogavero, Murano, and Vardi. This logic allows to reason explicitly about strategies as first order objects and express in a very natural and elegant way several solution concepts like Nash, resilient, and secure equilibria, dominant strategies, etc. The price that one has to pay for the high expressiveness of SL semantics is that agents strategies it admits may be not *behavioral*, i.e., a choice of an agent, at a given moment of a play, may depend on the choices another agent can make in another counterfactual play. As the latter moves are unpredictable, this kind of strategies cannot be synthesized in practice.**

In this paper, we investigate two syntactical fragments of SL, namely the *conjunctive-goal* and *disjunctive-goal*, called SL[CG] and SL[DG] for short, and prove that their semantics admit behavioral strategies only. These logics are obtained by forcing SL formulas to be only of the form of conjunctions or disjunctions of goals, which are temporal assertions associated with a binding of agents with strategies. As SL formulas with any Boolean combination of goals turn out to be non behavioral, we have that SL[CG] and SL[DG] represent the maximal fragments of SL describing agent behaviors that are synthesizable. As a consequence of the above results, the model-checking problem for both SL[CG] and SL[DG] is shown to be solvable in 2EXPTIME, as it is for the subsumed logic ATL*.

I. INTRODUCTION

In recent years, game theory has exhibited to be a fruitful metaphor in open-system verification, where the evolution emerges from the coordination of different parts viewed as autonomous and proactive agents [1], [2]. This has encouraged the development of several modal logics aimed at reasoning about strategies and their interaction [3]–[7].

An important contribution in this field has been the development of *Alternating-Time Temporal Logic* (ATL*, for short), introduced by Alur, Henzinger, and Kupferman [3]. Formally, it is obtained as a generalization of branching-time logic CTL* [8], where the path quantifiers *there exists* "E" and *for all* "A" are replaced with strategic modalities of the form "$\langle\langle A \rangle\rangle$" and "$[[A]]$", for a set A of *agents*. These modalities are used to express cooperation and competition among agents in order to achieve a temporal goal. Several decision problems have been investigated about ATL*. In particular, the model-checking problem is proved to be 2EXPTIME-COMPLETE [3].

Despite its powerful expressiveness, ATL* suffers from two strong limitations: 1) strategies are treated implicitly through modalities that refer to games between competing coalitions and 2) strategic modalities only represent coupled $\exists\forall$ and $\forall\exists$ quantifications over strategies. These limitations make ATL* less suited to formalize several important strategic notions, such as *Nash Equilibrium* and the like.

The above considerations led to introduce and study in [9] a more powerful logic for strategic reasoning in order to "unpack" and extend the ATL* modalities. The result is *Strategy Logic* (SL, for short), which extends ATL* in two fundamental aspects. First, strategies are first-order objects that can be existentially and universally quantified. Specifically, SL uses the existential $\langle\langle x \rangle\rangle$ and the universal $[[x]]$ strategic modalities, which can be read as *"there exists a strategy x"* and *"for all strategies x"*, respectively. Second, strategies represent general conditional plans that at each step prescribe an action on the base of the previous history. Thus, strategies are not intrinsically glued to a specific agent, but an explicit *binding* operator (a, x) allows to bind an agent a to the strategy associated with a variable x.

The SL features allow a finer-grained description of extensive games and reveals aspects that other classic subsumed logics, such as ATL*, are not able to grasp. In particular, a key aspect in the theory of extensive games is that strategies are *behavioral*, that is, by repeating Myerson's words: "the correlation between the move he [i.e., an agent] chooses at one information state and the move would have chosen at any other information state has no significance" [10].

However, the expressive power of SL allows to specify some sentences that can be satisfied only by agent strategies that are *not* behavioral. More specifically, in a determined history of the play, what Myerson calls "information state", the value of an existential quantified strategy may depend on how the other strategies will behave in the future or in other counterfactual plays. This means that, to choose an existential strategy, we need to know the entire structure of universal strategies on all possible histories, which is in general unpredictable, as what we actually know is their behavior on the history of interest only. This means that non-behavioral strategies cannot be synthesized in practice.

Additionally, by maintaining in SL this kind of strategies, we lose important model-theoretic properties and incur in an increased complexity of related decision problems. In particular, in [11], it has been shown that the model-checking problem becomes non-elementary complete. To gain back elementariness, several fragments of SL, strictly subsuming ATL*, have been investigated and studied in [11], [12]. Among the others, *One-Goal Strategy Logic* (SL[1G], for short)

[*]Partially supported by the FP7 European Union project 600958-SHERPA and the IndAM 2013 project "Logiche di Strategie Estese".

encompasses formulas in a special prenex normal form having a single temporal goal at a time. For a goal, it is specifically meant 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 a linear-time formula. 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. In [11], [12], it has been showed that SL[1G] admits behavioral strategies[1] only and this has been a key aspect in showing that the model-checking problem for this logic is 2EXPTIME-COMPLETE, as it is for ATL*. This adds motivations on the importance to restrict our attention to behavioral strategies in SL.

In this paper, we achieve the target of identifying sufficient criteria for what is behavioral in SL. Specifically, we consider two syntactical fragments of SL, strictly subsuming SL[1G], namely the *conjunctive-goal* and *disjunctive-goal*, respectively called SL[CG] and SL[DG], for short. These logics are obtained by forcing SL formulas to be only of the form of conjunctions or disjunctions of goals. As main result, we show that also SL[CG] and SL[DG] admit behavioral strategies only. Moreover, since SL formulas with any Boolean combination of goals, named SL[BG] in [11], [12], turn out to be non behavioral, we have that SL[CG] and SL[DG] represent the maximal syntactic fragments of SL that are behavioral. We also solve the model-checking problem for SL[CG] and SL[DG] via alternating tree automata. As for SL[1G], the fact that these fragments are behavioral is a key aspect to strongly simplify the reasoning about strategies by reducing it to a step-by-step reasoning about which action to perform. More precisely, we avoid the projection operation for each quantification in the formula (as it is instead required for SL) by using a dedicated automaton that makes an action quantification for each node of the tree model. As a formula may require different goals to take care of simultaneously, a specific duty of the automaton is to show that such goals can be treated separately, but maintaining their mutual coherence. As this automaton is only exponential in the size of the formula (in particular, it is independent from the alternation number of quantifications in the formula) and its non-emptiness can be computed in exponential time, we get that the model-checking for both SL[CG] and SL[DG] is solvable in 2EXPTIME. It is interesting to observe that the same result holds for SL[1G], while for SL[BG] it still open the question whether the model checking problem is elementary or not.

Besides the theoretical aspects, SL[CG] and SL[DG] enable to formalize several interesting game properties that cannot be expressed in SL[1G], and hence in ATL*. Specifically, on one hand, in SL[CG] it is possible to formalize scenarios where an agent can join two or more different coalitions without producing mutual conflicts or, somewhat conversely, it can assure that none of them prevails over the others. As it will be clear in the following, in such a case, conjunctive goals are strictly required. As shown in [13], this feature is essential also to address interesting issues such as coalition decomposability, that is whether a whole coalition can be split in smaller, and hence more manageable, sub-coalitions. On the other hand, in SL[DG] it is possible to formalize implication between goals, which can be used to express that a winning condition is weaker that another one.

Due to space limit, most of the proofs are omitted and reported in an extended version. We refer to [9], [11], [12] for more motivations, examples and related material. For other recent works in the field of strategic reasoning, one can also see [14]–[22].

Outline: The rest of the work is structured as follows. In Section II, we first introduce syntax and semantics of SL and introduce its fragments SL[CG] and SL[DG], along with some inspiring examples. Then, we give the notion of *dependence map*, which is used to define the concept of behavioral semantics. In Section III, we show that classical and behavioral semantics for both SL[CG] and SL[DG] coincides. In Section IV, we describe the model-checking procedure for the introduced fragments and show a 2EXPTIME upper bound. Finally, we conclude the work with some observations and discussions.

II. STRATEGY LOGIC

In this section, we first describe concurrent game structures and give some preliminary notions. Then, we recall syntax and semantics of Strategy Logic (SL, for short) and introduce the new fragments SL[CG] and SL[DG]. Finally, we define the notions of dependence maps and behavioral semantics. For a detailed introduction on SL, see [11].

A. Concurrent game structures

As in ATL*, SL is interpreted over *concurrent game structures* [3], which are *labeled transition system*,where each state represents a configuration of an extensive game characterized by a set of atomic propositions denoting its meaning. Transitions between states represent possible concurrent moves of the players involved in the game.

Definition II.1 (Concurrent Game Structures)**.** *A* concurrent game structure *(*CGS*, for short) is a tuple* $\mathcal{G} \triangleq \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}, \mathrm{St}, \lambda, \tau, s_0 \rangle$*, where* AP *and* Ag *are finite non-empty sets of* atomic propositions *and* agents*,* Ac *and* St *are enumerable non-empty sets of* actions *and* states*,* $s_0 \in \mathrm{St}$ *is a designated* initial state*, and* $\lambda : \mathrm{St} \to 2^{\mathrm{AP}}$ *is a* labeling function *that maps each state to the set of atomic propositions true in that state. Let* $\mathrm{Dc} \triangleq \mathrm{Ac}^{\mathrm{Ag}}$ *be the set of* decisions*, i.e., functions from* Ag *to* Ac *representing the choices of an action for each agent.* [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.*

[1]In [11], [12], we improperly call this kind of strategies *elementary*.

[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.

In the following, 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. Before introducing our logic, we need some preliminary definitions.

A *track* (resp., *path*) is a finite (resp., an infinite) sequence of states $\rho \in \mathrm{St}^*$ (resp., $\pi \in \mathrm{St}^\omega$) such that, for all $0 \leq i \leq |\rho| - 1$ (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., $(\pi)_{i+1} = \tau((\pi)_i, \mathsf{d})$). The set $\mathrm{Trk} \subseteq \mathrm{St}^+$ (resp., $\mathrm{Pth} \subseteq \mathrm{St}^\omega$) contains all non-zero length tracks (resp., paths). Moreover, $\mathrm{Trk}(s)$ (resp., $\mathrm{Pth}(s)$) indicates the subsets of tracks (resp., paths) starting at a state $s \in \mathrm{St}$. 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 history of the game.

A *strategy* is a partial function $\mathsf{f} : \mathrm{Trk} \rightharpoonup \mathrm{Ac}$ that maps each non-zero length track in its domain to an action. Intuitively, a strategy is a conditional plan that for each partial history of $\mathcal{G}$ prescribes an action to be executed. 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., $\mathsf{dom}(\mathsf{f}) = \mathrm{Trk}(s)$. The set $\mathrm{Str} \triangleq \mathrm{Trk} \rightharpoonup \mathrm{Ac}$ (resp., $\mathrm{Str}(s) \triangleq \mathrm{Trk}(s) \rightarrow \mathrm{Ac}$) contains all (resp., s-total) strategies.

Let $\mathsf{f} \in \mathrm{Str}$ be a strategy and ρ a track in its domain. Then, $(\mathsf{f})_\rho \in \mathrm{Str}$ denotes the *translation* of a strategy f along ρ, i.e. a strategy such that $(\mathsf{f})_\rho(\rho') \triangleq \mathsf{f}(\rho \cdot \rho'_{\geq 1})$, if $(\rho')_0 = \mathsf{lst}(\rho)$ and $\rho \cdot \rho'_{\geq 1} \in \mathsf{dom}(\mathsf{f})$, undefined otherwise. [3] Intuitively, the translation $(\mathsf{f})_\rho$ is the update of the strategy f, once the history of the game becomes ρ. It is worth noting that, if f is a $(\rho)_0$-total strategy then $(\mathsf{f})_\rho$ is $\mathsf{lst}(\rho)$-total.

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 $l \in \mathsf{dom}(\chi)$. The set $\mathrm{Asg} \triangleq \mathrm{Var} \cup \mathrm{Ag} \rightharpoonup \mathrm{Str}$ (resp., $\mathrm{Asg}(s) \triangleq \mathrm{Var} \cup \mathrm{Ag} \rightharpoonup \mathrm{Str}(s)$) contains all (resp., s-total) assignments. Moreover, $\mathrm{Asg}(\mathrm{X}) \triangleq \mathrm{X} \rightarrow \mathrm{Str}$ (resp., $\mathrm{Asg}(\mathrm{X}, s) \triangleq \mathrm{X} \rightarrow \mathrm{Str}(s)$) indicates the subset of X-*defined* (resp., s-total) assignments, i.e., (resp., s-total) assignments defined on $\mathrm{X} \subseteq \mathrm{Var} \cup \mathrm{Ag}$.

As in the case of strategies, we also define a *translation* along a given track for assignments. For a given state $s \in \mathrm{St}$, let $\chi \in \mathrm{Asg}(s)$ be an s-total assignment and $\rho \in \mathrm{Trk}(s)$ a track. Then, $(\chi)_\rho \in \mathrm{Asg}(\mathsf{lst}(\rho))$ denotes 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 $l \in \mathsf{dom}(\chi)$.

As in first order logic, in order to quantify over strategies or bind a strategy to an agent, we update an assignment χ by associating an agent or a variable l with a new strategy f. Let $\chi \in \mathrm{Asg}$ be an assignment, $\mathsf{f} \in \mathrm{Str}$ a strategy and $l \in \mathrm{Var} \cup \mathrm{Ag}$ either an agent or a variable. Then, $\chi[l \mapsto \mathsf{f}] \in \mathrm{Asg}$ denotes the new assignment defined on $\mathsf{dom}(\chi[l \mapsto \mathsf{f}]) \triangleq \mathsf{dom}(\chi) \cup \{l\}$ that returns f on l and the same value that χ would return on the rest of its domain. It is worth noting that if χ and f are s-total then $\chi[l \mapsto \mathsf{f}]$ is also s-total.

A *play* is the unique outcome of the game determined by all agent strategies participating to it; formally, given a state $s \in \mathrm{St}$ and a complete s-total assignment $\chi \in \mathrm{Asg}(s)$, the function $\mathsf{play}(\chi, s)$ returns the path $\pi \in \mathrm{Pth}(s)$ such that, 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 $a \in \mathrm{Ag}$.

Finally, for a state $s \in \mathrm{St}$ and a complete s-total assignment $\chi \in \mathrm{Asg}(s)$, we define the i-th global translation that calculates, at a certain step $i \in \mathbb{N}$ of the play, what is the current state and its updated assignment. Formally, the *i-th global translation* of (χ, s) is the pair of a complete assignment and a state $(\chi, s)^i \triangleq ((\chi)_{(\pi)_{\leq i}}, (\pi)_i)$, where $\pi = \mathsf{play}(\chi, s)$.

As in the case of components of a CGS, in order to avoid any ambiguity, we sometimes use the name of the CGS as a subscript of the sets and functions introduced above.

B. Syntax and semantics

Strategy Logic (SL, for short) [9] syntactically extends LTL with two *strategy quantifiers*, the existential $\langle\langle x \rangle\rangle$ and the universal $[[x]]$, and an *agent binding* (a, x), where a is an agent and x a variable. Intuitively, these new elements can be respectively read as *"there exists a strategy x"*, *"for all strategies x"*, and *"bind agent a to the strategy associated with the variable x"*.

Definition II.2 (SL Syntax). *Given the set of atomic propositions* AP*, variables* Var*, and agents* Ag*, the formal syntax of* SL *is defined as follows:*

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

where $p \in \mathrm{AP}$, $x \in \mathrm{Var}$, and $a \in \mathrm{Ag}$.

By $\mathsf{sub}(\varphi)$ we denote the set of subformulas of the formula φ. For instance, the subformulas of $\varphi = \langle\langle \mathsf{x} \rangle\rangle (a, \mathsf{x})(\mathsf{F}\,\mathsf{p})$ are $\mathsf{sub}(\varphi) = \{\varphi, (a, \mathsf{x})(\mathsf{F}\,\mathsf{p}), (\mathsf{F}\,\mathsf{p}), \mathsf{p}\}$.

Usually, predicative logics need the concepts of free and bound *placeholders* in order to formally define their semantics. For example, in first order logic the only type of placeholders are variables. In SL, since strategies can be associated to both agents and variables, we use the set of *free agents/variables* $\mathsf{free}(\varphi)$ 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.

[3]By $\mathsf{lst}(w) \triangleq (w)_{|w|-1}$ it is denoted the *last element* of a finite non-empty sequence $w \in \Sigma^+$. Moreover, the notations $(w)_{\leq i} \in \Sigma^*$ and $(w)_{>i} \in \Sigma^\infty$ indicate the *prefix* up to and the *suffix* from index $i \in [0, |w|[$ of a non-empty sequence $w \in \Sigma^\infty$.

The SL semantics is defined as follows.

Definition II.3 (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) *Boolean operators are interpreted as usual.*
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* $\mathcal{G}, (\chi, s)^j \models \varphi_1$.

It is evident that, due to Items 1, 2, and 3, the LTL semantics is simply embedded into the SL one. Furthermore, since the satisfaction of a sentence φ does not depend on assignments, we omit them and write $\mathcal{G}, s \models \varphi$, when s is a generic state in St, and $\mathcal{G} \models \varphi$ when $s = s_0$.

In what follows 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 $\wp$. By $\langle\langle \wp \rangle\rangle$ (resp. $[[\wp]]$) we denote the set of variables occurring *existentially* (resp. universally) quantified in $\wp$. A *binding prefix* $\flat$ over V is a word of type $(a_1, x_1) \ldots (a_m, x_m)$ such that $x_i \in \mathrm{V}$, for each $1 \leq i \leq m$, and $\mathrm{Ag} = \{a_1, \ldots, a_m\}$. By $\mathrm{Qnt}(\mathrm{V})$ and $\mathrm{Bnd}(\mathrm{V})$ we indicate the set of quantification and binding prefixes over V.

We are now ready to introduce the two fragments of SL we are interested in this paper, i.e., the *conjunctive-goal* and *disjunctive-goal strategy logic* (SL[CG] and SL[DG], for short). In the following, when we have to refer to one of the two logics indifferently, we use the single acronym SL[XG].

Definition II.4 (SL[XG] Syntax)**.** *The syntax of* SL[CG] *and* SL[DG] *is defined as follows, with the symbol* $\circledast$ *used in place of* $\wedge$ *or* $\vee$*, respectively:*

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

where $\wp \in \mathrm{Qnt}(\mathsf{free}(\psi))$ *and* $\flat \in \mathrm{Bnd}(\mathrm{V})$*, for a given set of variables* $\mathrm{V} \subseteq \mathrm{Var}$.

The name of these fragments comes from the fact that formulas of the kind $\flat\varphi$ are called goals. It is also evident that SL[XG] resides between the logics SL[1G] (see [12], for more), where ψ consists of a unique goal $\flat\varphi$, and SL[BG] (see [11], for more), which allows any Boolean combination of goals.

A first important result about SL[XG], is that, differently from SL[1G], it is not invariant under decision-unwinding. Intuitively, by decision-unwinding [12] we mean the tree obtained by unraveling a given CGS w.r.t. all possible agent decisions. Consequently, SL[XG] is strictly more expressive than SL[1G].

Theorem II.1 (Unwinding Variance)**.** *There exists an* SL[XG] *sentence* φ *and two models* $\mathcal{G}_1$ *and* $\mathcal{G}_2$ *with isomorphic decision-unwindings such that* $\mathcal{G}_1 \models \varphi$ *and* $\mathcal{G}_2 \not\models \varphi$.

C. Inspiring examples

We now give a few specification examples, in order to show the expressive power of the introduced fragments.

As first example, assume that one wants to check whether an agent-provider can serve with a unique policy the requests of two agent-clients and the behavior of one agent-client cannot influence the request of the other one. This property can be described by the SL[CG] sentence $\phi = \langle\langle \mathsf{x} \rangle\rangle \langle\langle \mathsf{y}_2 \rangle\rangle \langle\langle \mathsf{y}_3 \rangle\rangle [[\mathsf{z}]]((\mathsf{a}_1, \mathsf{x})(\mathsf{a}_2, \mathsf{y}_2)(\mathsf{a}_3, \mathsf{z})\mathsf{F}\, \mathsf{p} \wedge (\mathsf{a}_1, \mathsf{x})(\mathsf{a}_2, \mathsf{z})(\mathsf{a}_3, \mathsf{y}_3)\mathsf{F}\, \mathsf{q})$, where a_1 is the provider, a_2 and a_3 are the two clients, and $\mathsf{F}\, \mathsf{p}$ and $\mathsf{F}\, \mathsf{q}$ are the relative requests. Note that the ATL* formula $\phi_1 = \langle\langle \{\mathsf{a}_1, \mathsf{a}_2\} \rangle\rangle \mathsf{F}\, \mathsf{p} \wedge \langle\langle \{\mathsf{a}_1, \mathsf{a}_3\} \rangle\rangle \mathsf{F}\, \mathsf{q}$ is too weak for this aim, since according to it the strategies that a_1 may need to adopt in the two coalitions could be mutually inconsistent. Similarly, $\phi_2 = \langle\langle \{\mathsf{a}_1, \mathsf{a}_2, \mathsf{a}_3\} \rangle\rangle (\mathsf{F}\, \mathsf{p} \wedge \mathsf{F}\, \mathsf{q})$ does not assure that the agent-clients are independent, in the sense that no one of them can jeopardize the request of the other agent. In particular, we have that ϕ implies $\phi_1 \wedge \phi_2$, but the converse is not necessarily true. More generally, sentences like ϕ can be used in several contexts. In the field of multi-agent systems, they tell us that the agent a_1 has a strategy that enables to form two independent coalitions with a_2 and a_3, respectively, without embarking on a unique coalition formation process with both of them. In the context of formal verification, instead, such sentences can be used at different levels of analysis to test the fault-tolerance of a component w.r.t. the others or the robustness of an entire system, represented by a_1, to possible misuses of single users (a_2 and a_3).

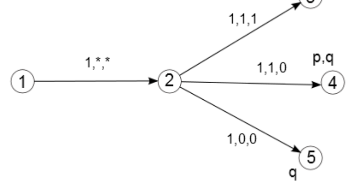

Figure 1. Provider with unique policy.

Now, consider the CGS $\mathcal{G}$ of Figure 1, where each transition is associated with a triple of binary values representing the decision of a_1, a_2, and a_3, respectively (self-loops have been omitted for simplicity). Let x and y_2 set to be the strategy that constantly executes the action 1 and y_3 set to be the strategy that constantly executes 0. It is easy to see that, from the initial state s_1, if x is bound to a_1

and y_2 is bound to a_2, then the system is forced to reach the states s_3 or s_4 (where p is true). Similarly, if x is bound to a_1 and y_3 is bound to a_3, the system is forced to reach the states s_4 or s_5 (where q is true). Thus, the previous sentence ϕ is satisfied in s_0.

As another example, let us consider a game with three agents a_1, a_2, and a_3 such that a_2 and a_3 want to achieve $\mathsf{G}\,\mathsf{F}\,\mathsf{p}$ and $\mathsf{G}\,\mathsf{F}\,\mathsf{q}$, respectively. We want to verify whether a_1 can act as a "super partes" agent that balances the evolution of the game so that the other agents can possibly satisfy their own goal. This can be expressed in SL[CG] by the sentence $\phi' = \langle\langle \mathsf{x}\rangle\rangle [[\mathsf{y}]] \langle\langle \mathsf{z}_1\rangle\rangle \langle\langle \mathsf{z}_2\rangle\rangle$ $((\mathsf{a}_1,\mathsf{x})(\mathsf{a}_2,\mathsf{y})(\mathsf{a}_3,\mathsf{z}_1)\mathsf{G}\,\mathsf{F}\,\mathsf{q} \quad\wedge\quad (\mathsf{a}_1,\mathsf{x})(\mathsf{a}_3,\mathsf{y})(\mathsf{a}_2,\mathsf{z}_2)\mathsf{G}\,\mathsf{F}\,\mathsf{p})$.

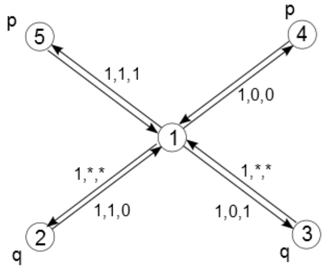

Figure 2. Fair scheduler.

Consider the CGS $\mathcal{G}$ of Figure 2. If a_1 constantly executes the action 1, then the even steps of each possible play will be the initial state s_1. From that state, a_2 cannot force the next state to yield $\neg\mathsf{q}$ and, vice versa, a_3 cannot force the next state to yield $\neg\mathsf{p}$. Thus, $\mathcal{G}$ satisfies ϕ' in s_1. It is worth noting that a_1 does not have the power to decide which winning condition will be satisfied, however by executing the strategy 0^ω it denies both of them.

As final issue, consider the problem to check whether, given a two-player CGS $\mathcal{G}$, a winning condition ψ_1 is more restrictive for a_1 that another condition ψ_2. An ATL* formula such as $[[\{\mathsf{a}_1\}]]\psi_1 \Rightarrow [[\{\mathsf{a}_1\}]]\psi_2$ could be used for this scope. A finer grained representation of the problem consists in analyzing the capability to obtain ψ_1 and ψ_2 for each single strategy of the other agent a_2: whenever a_1 is able to respond to a strategy of a_2 in order to achieve ψ_1, then, against that strategy, it can also achieve ψ_2. Formally, this can be obtained by the SL[DG] sentence $[[\mathsf{x}]][[\mathsf{y}]]\langle\langle\mathsf{z}\rangle\rangle((\mathsf{a}_2,\mathsf{x})(\mathsf{a}_1,\mathsf{y})\psi_1 \rightarrow (\mathsf{a}_2,\mathsf{x})(\mathsf{a}_1,\mathsf{z})\psi_2)$.

D. Strategy quantifications

In this section we formalize the behavioral semantics for the prenex fragment of SL, which plays a key role in maintaining the model-checking problem for SL[XG] as easy as ATL*. For this scope, we need first to define the concept of *dependence map* that is a Skolemization procedure for SL and, then, the related notion of *elementariness*, which represents a generic functional correspondent of what behavioral means for strategies. We start with some additional notion. Let $\wp \in \mathrm{Qnt}(\mathrm{V})$ be a quantification prefix over a set $\mathrm{V} \subseteq \mathrm{Var}$ of variables. For each $y \in \langle\langle\wp\rangle\rangle$, we use $\mathrm{Dep}(\wp, y)$ to denote the set of universally quantified variables $x \in [[\wp]]$ that precede y in $\wp$, that are the variable on which y depends. A *valuation* of variables over a set D is a partial function $\mathsf{v} : \mathrm{Var} \rightharpoonup \mathrm{D}$. By $\mathrm{Val}_{\mathrm{D}}(\mathrm{V}) \triangleq \mathrm{V} \rightarrow \mathrm{D}$ we denote the set of all valuation functions over D whose domain is V.

A *dependence map* for $\wp$ over D is a function $\theta : \mathrm{Val}_{\mathrm{D}}([[\wp]]) \rightarrow \mathrm{Val}_{\mathrm{D}}(\mathrm{V})$ satisfying the following properties: *(i)* $\theta(\mathsf{v})(x) = \mathsf{v}(x)$, for all $x \in [[\wp]]$ and *(ii)*, for all $\mathsf{v}_1, \mathsf{v}_2 \in \mathrm{Val}_{\mathrm{D}}([[\wp]])$ and $y \in \langle\langle\wp\rangle\rangle$, if $\mathsf{v}_{1\restriction \mathrm{Dep}(\wp,y)} = \mathsf{v}_{2\restriction \mathrm{Dep}(\wp,y)}$ then $\theta(\mathsf{v}_1)(y) = \theta(\mathsf{v}_2)(y)$, where $\mathsf{v}_{\restriction \mathrm{Dep}(\wp,y)}$ is the restriction of v to $\mathrm{Dep}(\wp, y)$. By $\mathrm{DM}_{\mathrm{D}}(\wp)$ we denote the set of all dependence maps of $\wp$ over D. Intuitively, Item (i) says 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 y in $\wp$ only depends on variables in $\mathrm{Dep}(\wp,y)$.

A fundamental theorem reported below states 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. This procedure, easily proved to be correct by induction on the structure of the formula in [11], can be seen as the equivalent of the *Skolemization* in first order logic. Here we give an intuition about it through an example. Consider the SL[XG] sentence $\varphi = \wp\psi$, where $\wp = [[\mathsf{x}]]\langle\langle\mathsf{y}\rangle\rangle$ and $\psi = ((\mathsf{a}_1,\mathsf{x})(\mathsf{a}_2,\mathsf{y})\mathsf{G}\,\mathsf{p}) \circledast ((\mathsf{a}_1,\mathsf{y})(\mathsf{a}_2,\mathsf{x})\mathsf{F}\,\neg\mathsf{p})$. In order to satisfy φ, we need to choose for every strategy associated with x a "right" strategy to associate with y such that ψ is true. A dependence map over strategies can be given to maintain the correlation between strategies for y chosen w.r.t. any possible one given for x.

Theorem II.2 (SL Strategy Quantification). *Let $\mathcal{G}$ be a* CGS *and $\varphi = \wp\psi$ be 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. Such a characterization enables the definition of alternative semantics, based on the choice of a subset of dependence maps that ensures better model properties and easier decision problems. Here, we consider the set of dependence maps that are *elementary*, which allows us to greatly simplify the reasoning about strategy quantifications by reducing them to a set of quantifications over actions, one for each track in their domains. Actually, this is a purely functional concept that allows to identify, from a computational point of view, a more tractable set of dependence maps over generic domain set.

We formally define elementariness, through the concept of *adjoint function*. To intuitively understand this concept, consider again the formula φ given above. As we have pointed out in Theorem II.2, a dependence map associates a strategy for y at every given strategy for x. By recalling that a strategy is a function from tracks to actions, this means that the behavior of the strategy for y on a certain track ρ, depends on the overall behavior of the strategy for x over all possible tracks. The adjoint function lets us to put all possible tracks as a common factor in the choice of strategies for y. Consequently, it will be enough for y to take into account how x behaves on ρ, that is the history of the current play. Let us now formalize the 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. 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 $t \in \mathrm{T}$. Thus, 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. Similarly, from an adjoint function it is possible to determine the original function unambiguously. Hence, there is 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 II.5 (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* $\mathrm{T} \to \mathrm{D}$.

At this point, as mentioned above, we introduce a notion of *behavioral satisfiability*, in symbols $\models_{\mathrm{B}}$, which requires the elementariness of dependence maps over strategies.

Definition II.6 (SL Behavioral Semantics)**.** *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,* $\mathcal{G} \models_{\mathrm{B}} \varphi$ *iff there exists a dependence map* $\theta \in \mathrm{EDM}_{\mathrm{Str}(s_0)}(\wp)$ *such that* $\mathcal{G}, \theta(\chi), s_0 \models \psi$*, for all* $\chi \in \mathrm{Asg}([\![\wp]\!], s_0)$.

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.

III. Behavioral Semantics

In this section, we show that in SL[XG] standard and behavioral semantics coincide. As seen in the previous section, the concept of elementariness is essentially a property of the dependence maps that allows to identify a way to satisfy a sentence in a behavioral manner. Looking at a prototypical SL[XG] sentence $\wp(\flat_1\psi_1 \circledast \cdots \circledast \flat_n\psi_n)$, with $\circledast$ being either $\wedge$ or $\vee$, a dependence map, once applied to the quantification prefix $\wp$, determines a set of strategies associated with the relative variables. Nevertheless, since each binding $\flat_i$ possibly "distributes" those strategies to the agents in a different way, we may have n different complete assignments for φ and, hence, as many evolutions of the game. For instance, in the example of Figure 1, if x, y_2, and y_3 are interpreted as before (i.e., 1^ω, 1^ω, 0^ω, respectively) and z is set to the strategy that constantly executes the action 1, then $\flat_1 = (\mathsf{a}_1, \mathsf{x})(\mathsf{a}_2, \mathsf{y}_2)(\mathsf{a}_3, \mathsf{z})$ produces the path $\pi_1 = \mathsf{s}_1\mathsf{s}_2\mathsf{s}_3\mathsf{s}_3 \cdots$ and $\flat_2 = (\mathsf{a}_1, \mathsf{x})(\mathsf{a}_2, \mathsf{z})(\mathsf{a}_3, \mathsf{y}_3)$ the path $\pi_2 = \mathsf{s}_1\mathsf{s}_2\mathsf{s}_4\mathsf{s}_4 \cdots$. Thus, the main difficulty w.r.t. the proof of behavioral semantics for SL[XG] consists in dealing with all these different evolutions coherently. However, the fact that the goals can only be put either in conjunction or in disjunction ensures that these evolutions of the game "run in parallel" without interfering with each other.

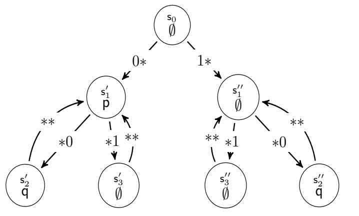
s_0 $\emptyset$
0∗
1∗
s'_1 p
s''_1 $\emptyset$
∗∗
∗0
∗1
∗∗
∗∗
∗1
∗0
∗∗
s'_2 q
s'_3 $\emptyset$
s''_3 $\emptyset$
s''_2 q

Figure 3. A CGS $\mathcal{G}$.

On the contrary, the same does not hold for SL[BG]. Consider, for example, the sentence $\varphi = [\![\mathsf{x}]\!]\langle\!\langle\mathsf{y}\rangle\!\rangle\langle\!\langle\mathsf{z}\rangle\!\rangle(\mathsf{a}, \mathsf{y})(\mathsf{b}, \mathsf{x})\mathsf{X}\,\mathsf{p} \leftrightarrow (\mathsf{a}, \mathsf{z})(\mathsf{b}, \mathsf{x})\mathsf{F}\,\mathsf{q}$ and the CGS $\mathcal{G}$ of Figure 3, where the agent a is the only one that controls the state s_0, while the agent b controls the states s'_1 and s''_1. It is not hard to see that $\mathcal{G} \models \varphi$. However, we have that $\mathcal{G} \not\models_{\mathrm{B}} \varphi$, since, to choose the value of the strategies to associate with the existential variables on s_0, we necessarily need to know the entire strategy associated to x and not only its value on s_0 itself. In particular, agent a should foresight whether agent b intends to achieve, at some point in the future of the states s'_1 and s''_1, the atomic proposition q or not. Therefore, agent a does not have any synthesizable action in the state s_0.

As in [11], to prove, instead, that SL[XG] admits behavioral strategies only, we reduce the check of $\mathcal{G} \models \varphi$ to a dependence-vs-valuation game, i.e., a turn-based *two player* game (TPG, for short), with the two players called *Even* and *Odd*, which is used to simulate the behavioral semantics. Intuitively, given a CGS $\mathcal{G}$ and an SL[XG] formula $\varphi = \wp\psi$, with $\psi = (\flat_1\psi_1 \circledast \cdots \circledast \flat_n\psi_n)$, a strategy for the player Even corresponds to an elementary dependence map, while a strategy for the player Odd corresponds to a valuation of the universal variables. Thus, $\mathcal{G} \models_{\mathrm{B}} \varphi$ iff Even has a strategy that, no matter how the player Odd sets the universal variables, forces the play to satisfy ψ.

A. Dependence-vs-valuation game

Before introducing the formalization of the dependence-vs-valuation game to which we reduce the verification of the SL[XG] modeling relation, we need the four auxiliary notions of *walk*, *evolution*, *dependence cluster*, and *valuation cluster* that constitute the building blocks of the game. In particular, evolutions are used as states of the game arena, while dependence and valuation clusters as possible moves of the two players. In addition, the walks allow to define the winning condition of the game. In the following, $\mathcal{G}$ denotes a CGS, $s \in \mathrm{St}$ a state, and $\wp \in \mathrm{Qnt}(\mathrm{V})$ and $\mathrm{B} \subseteq \mathrm{Bnd}(\mathrm{V})$, respectively, a quantification prefix and a set of binding prefixes for a given set of variables $\mathrm{V} \subseteq \mathrm{Var}$.

We have emphasized that, when the variables in V have been assigned with some strategies, different bindings produce in general different plays, so we introduce the notion of *walk* to formally deal with these multiple plays at the same time. Given a binding prefix $\flat \in \mathrm{B}$ we denote by $\zeta_\flat : \mathrm{Ag} \to \mathrm{V}$ the function associating with each agent the related variable in $\flat$, i.e., for all $a \in \mathrm{Ag}$, there is $0 \leq i < |\flat|$ such that $(\flat)_i = (a, \zeta_\flat(a))$. Then, a function $\mathsf{wlk} : \mathrm{B} \to \mathrm{Pth}_{\mathcal{G}}$ is a *walk* if there exists an assignment $\chi \in \mathrm{Asg}_{\mathcal{G}}(\mathrm{V}, s)$ such that $\mathsf{wlk}(\flat) = \mathsf{play}_{\mathcal{G}}(\chi \circ \zeta_\flat, s)$, for all $\flat \in \mathrm{B}$. By $\mathrm{Wlk}_{\mathcal{G}}(\mathrm{B}, s)$ we denote the set of all walks of $\mathcal{G}$ from s w.r.t. B. Moreover, by $\mathsf{wlk}_{\mathcal{G}}(\chi, \mathrm{B}, s) \in \mathrm{Wlk}_{\mathcal{G}}(\mathrm{B}, s)$ we indicate the unique walk derived from χ.

An *evolution* takes a *snapshot* at a certain step i of all tracks that are evolving in a walk, representing in this way a possible position of the verification game behind the check of $\mathcal{G} \models \varphi$. Formally, an evolution is a function $\mathsf{evl} : \mathrm{B} \to \mathrm{St}$ for which there exist a walk $\mathsf{wlk} \in \mathrm{Wlk}_{\mathcal{G}}(\mathrm{B}, s)$ and an index $i \in \mathbb{N}$ such that $\mathsf{evl}(\flat) = (\mathsf{wlk}(\flat))_i$, for all $\flat \in \mathrm{B}$. By $\mathrm{Evl}(\mathrm{B}, s)$ we denote the set of all evolutions of $\mathcal{G}$ from s w.r.t. B.

From now on, let $\mathsf{evl} \in \mathrm{Evl}(\mathrm{B}, s)$ be an evolution. Then, a *dependence cluster* w.r.t. evl is a function $\mathsf{dc} : \mathrm{B} \to \mathrm{DM}_{\mathrm{Ac}_{\mathcal{G}}}(\wp)$ such that, for all $\flat_1, \flat_2 \in \mathrm{B}$, if $\mathsf{evl}(\flat_1) = \mathsf{evl}(\flat_2)$ then $\mathsf{dc}(\flat_1) = \mathsf{dc}(\flat_2)$. By $\mathrm{DC}_{\mathcal{G}}(\wp, \mathrm{B}, \mathsf{evl}, s)$ we denote the set of all dependence clusters for $\mathcal{G}$ in s w.r.t. $\wp$, B, and evl. Intuitively, a dependence cluster represents, at a certain step, how a dependence map behaves along the tracks produced by different bindings. Since a dependence map produces a unique assignment of variables, if two bindings are going along the same track (which is given by the condition $\mathsf{evl}(\flat_1) = \mathsf{evl}(\flat_2)$), the dependence cluster has to return on them the same valuation of existential variables.

Finally, let V' a subset of V. Then, a *valuation cluster* w.r.t. evl and V' is a function $\mathsf{vc} : \mathrm{B} \to \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathrm{V}')$ such that, for all $\flat_1, \flat_2 \in \mathrm{B}$, if $\mathsf{evl}(\flat_1) = \mathsf{evl}(\flat_2)$ then $\mathsf{vc}(\flat_1) = \mathsf{vc}(\flat_2)$. By $\mathrm{VC}_{\mathcal{G}}(\mathrm{V}', \mathrm{B}, \mathsf{evl}, s)$ we denote the set of all valuation clusters for $\mathcal{G}$ in s w.r.t. V', B, and evl. Intuitively, a valuation cluster represents different valuations of universal variables of the quantification prefix along the tracks produced by different bindings. Like dependence clusters, as far as two bindings are running along the same track, valuations of universal variables have to be the same.

At this point, we define the required *dependence-vs-valuation game* that mimics the game behind the behavioral semantics of SL[XG]. Informally, it consists of *(i)* two non-empty non-intersecting sets of states, $\mathrm{N_e}$ and $\mathrm{N_o}$, for players *Even* and *Odd*, respectively, *(ii)* a designated initial state $n_0 \in \mathrm{N_e}$, and *(iii)* two binary relations $E_e \subseteq \mathrm{N_e} \times \mathrm{N_o}$ and $E_o \subseteq \mathrm{N_o} \times \mathrm{N_e}$ representing at each state the possible moves of Even and Odd, respectively. The notions of track, path, strategy, and play are analogous to those of a CGS and are not repeated here. However, to avoid confusion between concurrent game structures and the corresponding dependence-vs-valuation games, we denote with Mtc the set of all possible paths, namely *matches*, in a dependence-vs-valuation game. Finally, $\mathrm{Win} \subseteq \mathrm{Mtc}$ is the *winning set*, i.e., the set of all paths that are winning for Even.

Definition III.1 (Dependence-vs-Valuation Game)**.** *Let* $\mathrm{W} \subseteq \mathrm{Wlk}_{\mathcal{G}}(\mathrm{B}, s)$ *be a set of walks. Then, the dependence-vs-valuation game for* $\mathcal{G}$ *in* s *over* W *w.r.t.* $\wp$ *and* B *is the* TPG $\mathcal{H}(\mathcal{G}, s, \mathrm{W}, \wp, \mathrm{B}) \triangleq \langle \mathrm{N_e}, \mathrm{N_o}, E_e, E_o, n_0, \mathrm{Win} \rangle$ *where:*

- $\mathrm{N_e} \triangleq \mathrm{Evl}_{\mathcal{G}}(\mathrm{B}, s)$ *and* $\mathrm{N_o} \triangleq \{(\mathsf{evl}, \mathsf{dc}) : \mathsf{evl} \in \mathrm{Evl}_{\mathcal{G}}(\mathrm{B}, s) \wedge \mathsf{dc} \in \mathrm{DC}_{\mathcal{G}}(\wp, \mathrm{B}, \mathsf{evl}, s)\}$*;*
- $E_e \triangleq \{(\mathsf{evl}, (\mathsf{evl}, \mathsf{dc})) \in \mathrm{N_e} \times \mathrm{N_o}\}$ *and* $E_o \triangleq \{((\mathsf{evl}, \mathsf{dc}), \mathsf{evl}') \in \mathrm{N_o} \times \mathrm{N_e} : \exists \mathsf{vc} \in \mathrm{VC}_{\mathcal{G}}([\![\wp]\!], \mathrm{B}, \mathsf{evl}, s) \, s.t. \forall \flat \in \mathrm{B} . \mathsf{evl}'(\flat) = \tau(\mathsf{evl}(\flat), \mathsf{dc}(\flat)(\mathsf{vc}(\flat)) \circ \zeta_\flat)\}$*;*
- $n_0(\flat) \triangleq s$*, for all* $\flat \in \mathrm{B}$*;*
- $\mathrm{Win} \triangleq \{\varpi \in \mathrm{Mtc} : \exists \mathsf{wlk} \in \mathrm{W} . \forall i \in \mathbb{N}, \flat \in \mathrm{B} . (\varpi)_{2i}(\flat) = (\mathsf{wlk}(\flat))_i\}$.

Intuitively, the previous definition reformulates the verification process of a sentence φ as a zero-sum game where, step-by-step, the Even player attempts to satisfy φ by choosing "good" dependence clusters (Odd nodes) that, for each binding, interpret the existential variables and the Odd player replies with "bad" valuations of the universal variables (Even nodes) to try to make φ unsatisfied. In particular, the peculiarity of behavioral semantics is reformulated in this setting, through the concept of elementary dependence map, by means of the following notion of *encasement*.

Definition III.2 (Encasement)**.** *Let* $\mathrm{W} \subseteq \mathrm{Wlk}_{\mathcal{G}}(\mathrm{B}, s)$ *be a set of walks. Then,* W *is an* encasement *w.r.t.* $\wp$ *and* B *if there exists an elementary dependence map* $\theta \in \mathrm{EDM}_{\mathrm{Str}_{\mathcal{G}}(s)}(\wp)$ *such that, for all assignments* $\chi \in \mathrm{Asg}_{\mathcal{G}}([\![\wp]\!], s)$*, it holds that* $\mathsf{wlk}_{\mathcal{G}}(\theta(\chi), \mathrm{B}, s) \in \mathrm{W}$.

B. Proof of behavioral semantics

In this section, we describe the proof of the behavioral semantics for SL[XG], which allows us to construct, in the next section, a suitable automata-theoretic procedure for its model-checking problem. Note that, due to the length and high-complex nature of the whole proof, we only sketch its main steps and refer to the extended version for full details.

From a very high-level point of view, the proof schema proceeds in a similar way to what was done in [11] for SL[1G]. Therefore, in order to simplify the presentation and better understand the new proof, we first recall the structure of the old one: *(i)* we start constructing a two player game $\mathcal{H}$ simulating SL[1G] behavioral semantics of a given sentence $\phi = \wp\flat\psi$ over an assigned CGS $\mathcal{G}$; *(ii)* let P be the set of paths satisfying the LTL formula ψ, which is also the winning set of the game $\mathcal{H}$, we prove that it is an encasement iff $\mathcal{G} \models_{\mathrm{B}} \phi$ (see Definition 3.4 of [11]); *(iii)* via the encasement property, we show that if player Even wins $\mathcal{H}$ then P is an encasement and, vice versa, if P is Borelian but not an encasement then player Odd wins $\mathcal{H}$ (encasement characterization lemma, see Lemma B.5 of [11]); *(iv)* we prove that if player Odd wins the dual game $\overline{\mathcal{H}}$, obtained by the dual prefix $\overline{\wp}$, then player Even wins the original game $\mathcal{H}$ (dependence-vs-valuation duality lemma, see Lemma B.4 of [11]); *(v)* finally, we put all these facts together in the following way (see Theorem 4.24 of [11]): *(a)* we suppose that $\mathcal{G} \models \phi$ and assume $\overline{\mathrm{P}}$ as the set of paths satisfying $\neg\psi$, which is also the Borelian winning set of the dual game $\overline{\mathcal{H}}$; *(b)* since $\mathcal{G} \not\models_{\mathrm{B}} \neg\phi$, by Item *(ii)*, we get that $\overline{\mathrm{P}}$ is not an encasement; *(c)* thus, by Item *(iii)*, we derive that player Odd wins $\overline{\mathcal{H}}$; *(d)* so, by Item *(iv)*, we have that player Even wins the game $\mathcal{H}$; *(e)* now, again by Item *(iii)*, we get that P is an encasement; *(f)* finally, by a last application of Item *(ii)*, we conclude that $\mathcal{G} \models_{\mathrm{B}} \phi$. At the end, since the inverse direction, $\mathcal{G} \models_{\mathrm{B}} \phi$ implies $\mathcal{G} \models \phi$, is immediate by definition, we get that $\mathcal{G} \models \phi$ iff $\mathcal{G} \models_{\mathrm{B}} \phi$.

Clearly, due to the fact that an SL[XG] sentence has to deal with more than one play at a time, in order to apply the same proof structure to this logic, we have to generalize the two player game described above. Then, walks and evolutions allow to coherently track the verification process of temporal operators on all plays derived by the bindings, in one shot.

However, it is important to observe that parameterizing all two-player game components w.r.t. bindings does not entail by itself that the proof for SL[1G] works directly for SL[XG]. Indeed, another fundamental aspects in our new approach is to show that when one of the two players has a winning strategy, he has a decoupled winning strategy, i.e., a way to chose his moves w.r.t. a given binding that does not depend on what the other player has done before on bindings that are not running on the same track. To formally prove this fact, we first introduce a *topology* of *open* and *closed* sets on walks and then prove the following two preliminary facts: *(i)* the set of walks derived by a SL[XG] sentence is a member of the topology; *(ii)* if a player wins the two player game with an open/closed winning set then he has a decoupled winning strategy. Informally, a set of walks W is open if, for each walk $\mathsf{wlk} \in \mathrm{W}$, there exists a binding b for which the play $\mathsf{wlk}(b)$, whatever it is combined with other plays for the other bindings, form a new walk still contained in W. On the contrary, a set of walks W is closed if every walk wlk obtained by the shuffle of the plays of a set of walks $\{\mathsf{wlk}_1, \ldots \mathsf{wlk}_n\}$ in W is still a member of W. With this results as a tool, we are able to show the following generalizations of Lemmas B.5 and B.4 of [11], respectively.

Lemma III.1 (Encasement Characterization)**.** *Let* $\mathrm{W} \subseteq \mathrm{Wlk}_{\mathcal{G}}(\mathrm{B},s)$ *be an open/closed set of walks and* $\mathcal{H}{=}\langle\mathcal{A}, \mathrm{Win}\rangle = \mathcal{H}(\mathcal{G},s,\mathrm{W},\wp,\mathrm{B})$ *be the dependence-vs-valuation game for* $\mathcal{G}$ *in* s *over* W *w.r.t.* $\wp$ *and* B*. Then, the following hold:*

1) *if player Even wins* $\mathcal{H}$ *then* W *is an encasement w.r.t.* $\wp$ *and* B*;*
2) *if* Win *is a Borelian set and* W *is not an encasement w.r.t.* $\wp$ *and* B *then player Odd wins* $\mathcal{H}$*.*

Lemma III.2 (Dependence-vs-Valuation Duality)**.** *Let* $\mathrm{W} \subseteq \mathrm{Wlk}_{\mathcal{G}}(\mathrm{B}, s)$ *be an open/closed set of walks,* $\mathcal{H} = \langle\mathcal{A}, \mathrm{Win}\rangle = \mathcal{H}(\mathcal{G}, s, \mathrm{W}, \wp, \mathrm{B})$ *the dependence-vs-valuation game for* $\mathcal{G}$ *in* s *over* W *w.r.t.* $\wp$ *and* B*, and* $\overline{\mathcal{H}} = \langle\overline{\mathcal{A}}, \mathrm{Mtc} \setminus \mathrm{Win}\rangle = \mathcal{H}(\mathcal{G}, s, \mathrm{Wlk}_{\mathcal{G}}(\mathrm{B}, s) \setminus \mathrm{W}, \overline{\wp}, \mathrm{B})$ *its dual game. Then, if player Odd wins the dual* TPG $\overline{\mathcal{H}}$*, player Even wins the* TPG $\mathcal{H}$*.*

At this point, we can sketch the proof of behavioral semantics for SL[CG] (resp., SL[DG]): *(a)* we suppose that $\mathcal{G} \models \phi$, where $\phi = \wp \bigwedge_{\flat\in\mathrm{B}} \flat\psi_\flat$ (resp., $\phi = \wp \bigvee_{\flat\in\mathrm{B}} \flat\psi_\flat$) and assume $\overline{\mathrm{W}}$ as the set of walks satisfying $\neg \bigwedge_{\flat\in\mathrm{B}} \flat\psi_\flat$ (resp., $\neg \bigvee_{\flat\in\mathrm{B}} \flat\psi_\flat$), which is linked one-to-one to the Borelian winning set $\mathrm{Mtc}\setminus\mathrm{Win}$ of the dual game $\overline{\mathcal{H}}$; *(b)* we show that $\overline{\mathrm{W}}$ is an open (resp., closed) set, so, $\mathrm{W} \triangleq \mathrm{Wlk}_{\mathcal{G}}(\mathrm{B}, s) \setminus \overline{\mathrm{W}}$ is closed (resp., open); *(c)* since $\mathcal{G} \not\models_{\mathrm{B}} \neg\phi$, we obtain that $\mathrm{Mtc} \setminus \mathrm{Win}$ is not an encasement; *(d)* therefore, by Item 2 of Lemma III.1, we derive that player Odd wins $\overline{\mathcal{H}}$; *(e)* consequently, by Lemma III.2, we have that player Even wins the original game $\mathcal{H}$; *(f)* now, by Item 1 of Lemma III.1, we obtain that the set of walks W, from which $\mathcal{H}$ is derived, is an encasement; *(g)* finally we conclude that $\mathcal{G} \models_{\mathrm{B}} \phi$.

Theorem III.1 (SL[XG] Behavioral Semantics)**.** *For all* SL[XG] *sentences* φ*, it holds that* $\mathcal{G} \models \varphi$ *iff* $\mathcal{G} \models_{\mathrm{B}} \varphi$*.*

Finally, note that the same two player game we have defined for SL[XG] could also be used to prove the behavioral semantics of the full SL[BG], as its structure does not change when we have generic Boolean combinations of goals. However, since we already know that SL[BG] admits non-behavioral strategies too, an immediate question that promptly arises is why the proof does not work for the latter logic. Essentially, the problem is that the winning set derived from a SL[BG] sentence does not satisfy the decoupled property we mentioned above. This is due to the fact that the induced sets of walks are neither open nor closed, but belong to higher levels of the related topological hierarchy. So, Lemmas III.1 and III.2 cannot hold for SL[BG].

IV. Model-Checking Procedure

We finally solve the model-checking problem for SL[XG] and show that it is 2EXPTIME-COMPLETE, as it is for the less expressive ATL* and SL[1G] logics. The algorithmic procedure is based on an automata-theoretic approach, which reduces the decision problem for our logic to the emptiness problem of a suitable universal Co-Büchi tree automaton (UCT, for short) [23]. Our technique is innovative w.r.t. those proposed in literature for CTL* [24] and ATL* [3], since it is based on the novel notion of elementary dependence map and behavioral semantics. In particular, we extend the procedure proposed in [11] for SL[1G], along with a machinery to handle the conjunction/disjunction of goals. The high-level idea behind this approach is to avoid the use of projections for the strategy quantifications (which is instead required for SL), by reducing them to action quantifications, which can be managed on each state of the model without a non-elementary blow-up. Naturally, this approach is correct, since SL[XG] has behavioral strategies only, as we proved before.

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

Intuitively, this is a tree $\mathcal{T}_\chi$ whose nodes correspond to all possible histories in the unraveling of the CGS $\mathcal{G}$ and whose labeling represents, for the given assignment χ, all actions that the strategies associated to the variables prescribe.

Definition IV.1 (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 $\mathcal{T}_\chi \triangleq \langle \mathrm{T}, \mathsf{u} \rangle$, where $\mathrm{T} \triangleq \{\rho_{\geq 1} : \rho \in \mathrm{Trk}_\mathcal{G}(s)\}$, is an* assignment-state encoding *for χ if it holds that $\mathsf{u}(t) \triangleq (\widehat{\chi}(s \cdot t), \mathsf{lst}(s \cdot t))$, for all $t \in \mathrm{T}$.*

By a suitably embedding of the Vardi-Wolper construction [25] into a tree automaton, we build an UCT recognizing all assignment-state encodings derived by assignments satisfying a given goal.

Lemma IV.1 (Goal Automaton). *Let $\mathcal{G}$ be a* CGS *and $\flat\psi$ a goal without principal subsentences. There is a* UCT *$\mathcal{U}^\mathcal{G}_{\flat\psi}$ with $\mathtt{O}(2^{|\psi|})$ states 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 $\mathcal{G}, \chi, s \models \flat\psi$ iff $\mathcal{T}_\chi \in \mathrm{L}(\mathcal{U}^\mathcal{G}_{\flat\psi})$, where $\mathcal{T}_\chi$ is the assignment-state encoding for χ and $\mathrm{L}(\mathcal{U}^\mathcal{G}_{\flat\psi})$ is the set of trees $\mathcal{U}^\mathcal{G}_{\flat\psi}$ accepts.*

Now, we introduce an encoding for the information contained into the elementary dependence maps over strategies used to satisfy a given sentence. Intuitively, this is a tree similar to the assignment-state encoding, except that the labeling is used to represent the dependence maps over actions contained into the dependence map over strategies.

Definition IV.2 (Elementary Dependence-State Encoding). *Let $\mathcal{G}$ be a* CGS, *$s \in \mathrm{St}_\mathcal{G}$ one of its states, and $\theta \in \mathrm{EDM}_{\mathrm{Str}_\mathcal{G}(s)}(\wp)$ an elementary dependence map over strategies for a quantification prefix $\wp \in \mathrm{Qnt}(\mathrm{V})$ over the set $\mathrm{V} \subseteq \mathrm{Var}$. Then, a $(\mathrm{DM}_{\mathrm{Ac}_\mathcal{G}}(\wp) \times \mathrm{St}_\mathcal{G})$-labeled $\mathrm{St}_\mathcal{G}$-tree $\mathcal{T}_\theta \triangleq \langle \mathrm{T}, \mathsf{u} \rangle$, where $\mathrm{T} \triangleq \{\rho_{\geq 1} : \rho \in \mathrm{Trk}_\mathcal{G}(s)\}$, is an* elementary dependence-state encoding *for θ if it holds that $\mathsf{u}(t) \triangleq (\widetilde{\theta}(s \cdot t), \mathsf{lst}(s \cdot t))$, for all $t \in \mathrm{T}$.*

In the next lemma, we show the existence of an UCT that accepts a given elementary dependence-state encoding $\mathcal{T}$ iff an input UCT accepts all assignment-state encodings $\mathcal{T}'$ derived from $\mathcal{T}$. This automaton is used to handle the strategy quantifications on each state of the model, by means of quantification over actions modeled by the choice of an action dependence map.

Lemma IV.2 (Dependence Map Automaton). *Let $\mathcal{G}$ be a* CGS, *$\mathcal{U}$ a* UCT, *and $\wp \in \mathrm{Qnt}(\mathrm{V})$ a quantification prefix over $\mathrm{V} \subseteq \mathrm{Var}$. There is a* UCT *$\mathcal{U}_\wp$ with the same states of $\mathcal{U}$ such that, for all states $s \in \mathrm{St}_\mathcal{G}$ and elementary dependence maps over strategies $\theta \in \mathrm{EDM}_{\mathrm{Str}_\mathcal{G}(s)}(\wp)$, it holds that $\mathcal{T}_\theta \in \mathrm{L}(\mathcal{U}_\wp)$ iff $\mathcal{T}_{\theta(\chi)} \in \mathrm{L}(\mathcal{U})$, for all $\chi \in \mathrm{Asg}_\mathcal{G}([\![\wp]\!], s)$, where $\mathcal{T}_\theta$ and $\mathcal{T}_{\theta(\chi)}$ are, respectively, the elementary dependence-state encoding for θ and the assignment-state encoding for $\theta(\chi)$.*

We can now state the next theorem that is at the base of the model-checking procedure for SL[XG]. Actually, we build an automaton for an SL[CG] sentence and translate the model-checking question to its non-emptiness problem. In the case we start with an SL[DG] sentence, first we dualize it in an SL[CG] one, apply the previous construction, and check for the emptiness of the obtained automaton. Thus, the result for SL[DG] follows merely as a corollary.

Theorem IV.1 (SL[XG] Sentence Automaton). *Let $\mathcal{G}$ be a* CGS, *$s \in \mathrm{St}_\mathcal{G}$ one of its states, and $\phi = \wp \bigwedge_{\flat \in \mathrm{B}} \flat\psi_\flat$ (resp., $\phi = \wp \bigvee_{\flat \in \mathrm{B}} \flat\psi_\flat$) an* SL[CG] *(resp.,* SL[DG]*) sentence. Then, there exists an* UCT *$\mathcal{U}^{\mathcal{G},s}_\phi$ with $\mathtt{O}(2^{|\phi|})$ states such that $\mathcal{G}, \varnothing, s \models \phi$ iff $\mathrm{L}(\mathcal{U}^{\mathcal{G},s}_\phi) \neq \emptyset$ (resp., $\mathrm{L}(\mathcal{U}^{\mathcal{G},s}_\phi) = \emptyset$).*

Finally, by a simple calculation of the size of $\mathcal{U}_\phi$ and the complexity of the related (non) emptiness problem, we state in the next theorem the exact complexity of the model-checking problem for SL[XG]. Note that, in order to maintain a low data complexity, we first translate the UCT into an NPT and then make the product with the CGS under analysis.

Theorem IV.2 (SL[XG] Model-Checking). *The model-checking problem for* SL[XG] *is* PTIME-COMPLETE *w.r.t. the size of the model and* 2EXPTIME-COMPLETE *w.r.t. the size of the specification.*

V. DISCUSSION

Reasoning implicitly about strategies, as done in ATL* and its several variants, works well for two persons/teams zero-sum games, where at the abstract level what matters are simply the outcomes that players can ensure. However, for multi-player games with non-zero-sum objectives, it has been observed that an explicit representation of strategies can be useful in several cases to better capture the involved reasoning and in expressing various solution concepts, such as Nash, resilient, and secure equilibria. It is for this reason that a first-order framework such as SL, which allows quantification over strategy terms, is a natural setting to explore. Given this context, SL seems somehow even too expressive in the sense that it is possible to formulate sentences that can be satisfied only if we accept non-behavioral existentially quantified strategies. Such sentences can be considered as the *undesirable* part of SL for several reasons. First, their satisfaction joins together different possible plays of a game, whereas in standard game theory each single play ends up with an utility profile or a winning agent, so they do not have a clear game-theoretical counterpart. Second, their meaning is somewhat deceptive: even when they say that an agent can in principle respond to another agent, actually the relative strategy is not synthesizable, since it requires information that is inherently not at the disposal of that agent. Finally, also from a computational point of view, all relative reasoning tasks are severely affected. In particular, model checking is non-elementary in the size of the specification.

Clearly, all these considerations arise the problem to isolate the *desirable* part of SL that requires behavioral strategies only. This property essentially says that in order to reason about strategies it suffices to reason about individual actions, for each history of the game. Thanks to this, we are able to employ an automata-theoretic approach through which the strategy quantifications can be handled without the need to perform the projection operations, which usually implies a non-elementary blow up of the resulting automaton.

In this paper, we have addressed the question of which are the SL syntactic fragments of this sort. Specifically, we have introduced and investigated the conjunctive-goal and disjunctive-goal SL, respectively called SL[CG] and SL[DG], and shown that they admit a behavioral semantics. Since by allowing any Boolean combination of goals, the resulting logic, named SL[BG], does not retain this property [11], we have that SL[CG] and SL[DG] are the maximal syntactic fragments of SL that are behavioral. Moreover, we proved that their model-checking problem is 2EXPTIME-COMPLETE, i.e., asymptotically not more expensive than the one for SL[1G], which strictly subsumes ATL*. This strengthens the hypothesis that being behavioral is a sufficient condition to maintain the same complexity of ATL*.

Finally, it is important to stress that the fragments we have introduced in this work can be considered as the first relevant attempt to define a logic formalism able to describe synthesizable properties of non-zero-sum games, such as coalition decomposability, arbitrage and implication of winning conditions, etc., still maintaining a 2EXPTIME-COMPLETE model-checking problem.

ACKNOWLEDGMENTS

We thank G. De Giacomo and M.Y. Vardi for useful discussions on the concept of behavioral strategies.

REFERENCES

[1] E. Clarke, O. Grumberg, and D. Peled, *Model Checking.* MIT Press, 2002.

[2] O. Kupferman, M. Vardi, and P. Wolper, “Module Checking.” *IC*, vol. 164, no. 2, pp. 322–344, 2001.

[3] R. Alur, T. Henzinger, and O. Kupferman, “Alternating-Time Temporal Logic.” *JACM*, vol. 49, no. 5, pp. 672–713, 2002.

[4] S. Ghosh and R. Ramanujam, “Strategies in Games: A Logic-Automata Study.” in *ESSLLI'11*, ser. LNCS 7388. Springer, 2011, pp. 110–159.

[5] W. Jamroga and W. van der Hoek, “Agents that Know How to Play.” *FI*, vol. 63, no. 2-3, pp. 185–219, 2004.

[6] W. Jamroga and W. Penczek, “Specification and Verification of Multi-Agent Systems.” in *ESSLLI'11*, ser. LNCS 7388. Springer, 2011, pp. 210–263.

[7] F. Mogavero, A. Murano, and M. Vardi, “Relentful Strategic Reasoning in Alternating-Time Temporal Logic.” in *LPAR'10*, ser. LNAI 6355. Springer, 2010, pp. 371–387.

[8] E. Emerson and J. Halpern, "“Sometimes” and “Not Never” Revisited: On Branching Versus Linear Time.” *JACM*, vol. 33, no. 1, pp. 151–178, 1986.

[9] F. Mogavero, A. Murano, and M. Vardi, “Reasoning About Strategies.” in *FSTTCS'10*, ser. LIPIcs 8, 2010, pp. 133–144.

[10] R. Myerson, *Game Theory: Analysis of Conflict.* Harvard University Press, 1991.

[11] F. Mogavero, A. Murano, G. Perelli, and M. Vardi, “Reasoning About Strategies: On the Model-Checking Problem.” arXiv, Tech. Rep. 1112.6275, December 2011.

[12] ——, “What Makes ATL* Decidable? A Decidable Fragment of Strategy Logic.” in *CONCUR'12*, ser. LNCS 7454. Springer, 2012, pp. 193–208.

[13] G. Boella, L. Sauro, and L. van der Torre, “Strengthening Admissible Coalitions.” in *ECAI'06*. IOS Press, 2006, pp. 195–199.

[14] T. Agotnes, V. Goranko, and W. Jamroga, “Alternating-Time Temporal Logics with Irrevocable Strategies.” in *TARK'07*, 2007, pp. 15–24.

[15] T. Agotnes and D. Walther, “A Logic of Strategic Ability Under Bounded Memory.” *JLLI*, vol. 18, no. 1, pp. 55–77, 2009.

[16] B. Aminof, A. Legay, A. Murano, O. Serre, and M. Vardi, “Pushdown module checking with imperfect information.” *IC*, vol. 223, pp. 1–17, 2013.

[17] T. Brihaye, A. D. C. Lopes, F. Laroussinie, and N. Markey, “ATL with Strategy Contexts and Bounded Memory.” in *LFCS'09*, ser. LNCS 5407. Springer, 2009, pp. 92–106.

[18] K. Chatterjee, T. Henzinger, and N. Piterman, “Strategy Logic.” *IC*, vol. 208, no. 6, pp. 677–693, 2010.

[19] B. Finkbeiner and S. Schewe, “Coordination Logic.” in *CSL'10*, ser. LNCS 6247. Springer, 2010, pp. 305–319.

[20] A. D. C. Lopes, F. Laroussinie, and N. Markey, “ATL with Strategy Contexts: Expressiveness and Model Checking.” in *FSTTCS'10*, ser. LIPIcs 8, 2010, pp. 120–132.

[21] S. Pinchinat, “A Generic Constructive Solution for Concurrent Games with Expressive Constraints on Strategies.” in *ATVA'07*, ser. LNCS 4762. Springer, 2007, pp. 253–267.

[22] D. Walther, W. van der Hoek, and M. Wooldridge, “Alternating-Time Temporal Logic with Explicit Strategies.” in *TARK'07*, 2007, pp. 269–278.

[23] E. Grädel, W. Thomas, and T. Wilke, *Automata, Logics, and Infinite Games: A Guide to Current Research.*, ser. LNCS 2500. Springer, 2002.

[24] O. Kupferman, M. Vardi, and P. Wolper, “An Automata Theoretic Approach to Branching-Time Model Checking.” *JACM*, vol. 47, no. 2, pp. 312–360, 2000.

[25] M. Vardi and P. Wolper, “An Automata-Theoretic Approach to Automatic Program Verification.” in *LICS'86*. IEEE Computer Society, 1986, pp. 332–344.