# Logics for Multi-Agent Systems Verification


**Giuseppe Perelli**

Università degli studi di Napoli "Federico II"
Dipartimento di Matematica e Applicazioni "R. Caccioppoli"

Dottorato in **Scienze Computazionali e Informatiche**
*Ciclo XXVII*

A thesis submitted in fulfillment of the degree of
*Doctor in Compute Science*

Submission: March 31, 2015
Defense: Napoli, to be added
Revised version: March 25, 2015

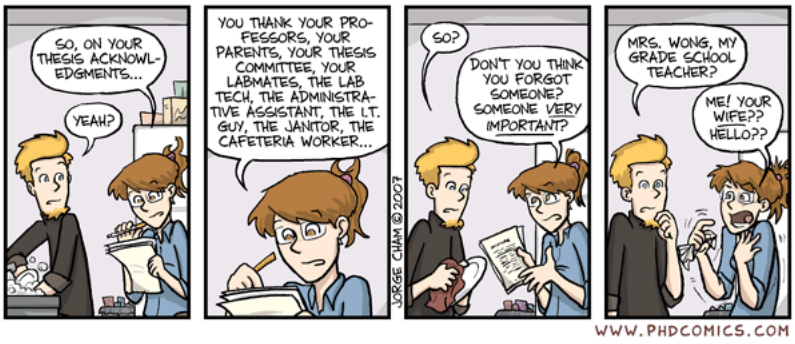
Supervisor: Prof. Dr. Aniello Murano

This paper intentionally left blank

“A problem is a chance for you to do your best.”
Duke Ellington

# Acknowledgments

The approaching end of a period easily evokes a final summary. In the case of a period ending with a thesis, such summary becomes mandatory. Otherwise, you risk the readers to complain for missing this fundamental chapter. It is indeed not a coincidence that the acknowledgements chapter is the most read and carefully revised by many many reviewers, the most severe you can ever meet. If missing a citation in the references of a paper sometime makes the difference between acceptance and rejection, missing a mention in the acknowledgements of a thesis always makes the difference between friendship and enmity.




First of all, I wish to say thanks to my **family**, for their constant presence, patience, and for all the moral support they never denied to me in these years. For having let me go alone unconditionally with my attitudes and ambitions, giving me the possibility to undertake an excellent path of knowledge and reach the goal of a PhD.

A big thank to my supervisor, Prof. **Aniello Murano**, for having perfectly interpreted the role of *academic parent*. Never ending source of good advices, he has been so good in wisely dosing the rations of encouragements, congrats, and scoldings, which has built a deep relation of trust and friendship along these years. With great emotion, I say thanks also to my *academic grandparent* (as he likes to define himself!), Prof. **Moshe Vardi**. The period working at Rice University under his guidance has been of big personal and professional growth. In every meeting with him, I am always amazed by his wise visions of research and life in general. The way he takes care of all his students, with sympathy and affection, is priceless! To complete the picture of my academic family, I would say thanks to my elder brother **Fabio Mogavero**. Every scientific conversation with him is always inspired and deep. The born understanding between us when we do research is extraordinary. Still today I am stunned by our ability to reason in harmony on all the problems we meet working together. Moreover, I would say thanks to all the professors and researchers I have had the pleasure to work with during these years: **Orna Kupferman**, **Angelo Montanari**, **Adriano Peron**, and **Luigi Sauro**.

For what the friends regards, I cannot avoid to mention all the ones that have been actively part of my PhD life, in particular in Houston. My officemate **Kuldeep**, for the philosophical conversations, the black humor jokes, the biscuits, and lunches full of sriracha. As well as for having taught me some useful hindi word ... Friends from the research group **Dror**, **Lucas**, **Miguel**, **Morteza**, and **Sumit**, for the fruitful collaboration and for all the suggestions. All the

Duncan Hall friends **Alina**, **Dragos**, **Ryan**, **Sailesh**, **Sarah**, and **Ted**, which whom I shared my spare time inside and outside campus.

A special mention to my Italian friends **Roberto**, **Stefania** and **Andrea**, for having let me find a bit of Naples and Italy thousands of kilometers far away from home. Our casual meeting in a supermarket on the other side of world has made us feel immediately tuned together. I still remember my crazy walk of about 7 miles to reach them the first time we hanged out. It is like I already knew they deserve my efforts! I will never forget all the dinner invitations, the bike (and walk!) adventures, and all the problems they have helped me to fix. A special remark for the two weeks in which **Roberta** joined us. After coming back to Naples, we started sharing a coffee from time to time, ant it is always a lot of fun. Last but not least thanks to **Rishi**, for the great sense of friendship and generosity they make him unique, and for having taught me the best hindi words to have success with Indian girls!

As some reader would have noticed, it seems I forgot to mention someone: the most severe reviewers of this chapter. This is clearly not true. In old Latin we say *dulcis in fundo* (sweetness in the end), and I would never forget the "women of my life" [0]: **Loredana** and **Sonali**. I met Loredana, three years ago. We started together the PhD adventure and our "colleagueship" immediately turned to be friendship. Our complicity has been very fundamental to effort both the office time and the several schools and conferences we participated together. Having such a special company, especially in the hard moments, would be a big fortune for everyone. Sonali has filled my days in Houston with her sweetness and happiness. Her genuine affection towards me makes me in a good mood just with a simple smile. Due to the long distance, we made an agreement: to meet every at most 3 years. I will do the best to respect this agreement, in order to be present in all the most important moments of our lives.

[0] I can still afford to have more than one!

# Contents

# Introduction

In system design, *model checking* is a well-established formal method that allows to automatically check for global system correctness [CE81, QS81, CGP02]. In such a framework, in order to check whether a system satisfies a required property, we describe its structure in a mathematical model (such as *Kripke structures* [Kri63] or *labeled transition systems* [Kel76]), specify the property with a formula of a temporal logic (such as LTL [Pnu77], CTL [CE81], or CTL$^\star$ [EH86]), and check formally that the model satisfies the formula. In the last decade, interest has arisen in analyzing the behavior of individual components or sets of them in systems with several entities. This interest has started in reactive systems, which are systems that interact continuously with their environments. In *module checking* [KVW01], the system is modeled as a module that interacts with its environment and correctness means that a desired property holds with respect to all such interactions (see also [FMP08] and [ALM$^+$13, JM14] for recent works in this field).

Starting from the study of module checking, researchers have looked for logics focusing on the strategic behavior of agents in multi-agent systems [ÅGJ07, AHK02, Pau02, JvdH04, WvdHW07, BJ14]. One of the most important developments in this field is *Alternating-Time Temporal Logic* (ATL$^\star$, for short), introduced by Alur, Henzinger, and Kupferman [AHK02]. ATL$^\star$ allows reasoning about strategies of agents with temporal goals. Formally, it is obtained as a generalization of CTL$^\star$ in which the path quantifiers, *there exists* "$\mathtt{E}$" and *for all* "$\mathtt{A}$", are replaced with *strategic modalities* of the form "$\langle\!\langle \mathrm{A} \rangle\!\rangle$" and "$[\![ \mathrm{A} ]\!]$", where $\mathrm{A}$ is a set of *agents* (a.k.a. *players*). Strategic modalities over agent sets are used to express cooperation and competition among them in order to achieve certain goals. In particular, these modalities express selective quantifications over those paths that are the result of infinite games between a coalition and its complement.

ATL$^\star$ formulas are interpreted over *concurrent game structures* (CGS, for short) [AHK02], which model interacting processes. Given a CGS $\mathcal{G}$ and a set $\mathrm{A}$ of agents, the ATL$^\star$ formula $\langle\!\langle \mathrm{A} \rangle\!\rangle \psi$ is satisfied at a state $s$ of $\mathcal{G}$ if there is a set of strategies for agents in $\mathrm{A}$ such that, no matter which strategies are executed by agents not in $\mathrm{A}$, the resulting outcome of the interaction in $\mathcal{G}$ satisfies $\psi$ at $s$. Thus, ATL$^\star$ can express properties related to the interaction among components, while CTL$^\star$ can only express property of the global system. As an example, consider the property "processes $\alpha$ and $\beta$ cooperate to ensure that a system (having more than two processes) never enters a failure state". This can be expressed by the ATL$^\star$ formula $\langle\!\langle \{\alpha, \beta\} \rangle\!\rangle \mathtt{G} \neg \mathit{fail}$, where $\mathtt{G}$ is the classical LTL temporal operator "*globally*". CTL$^\star$, in contrast, cannot express this property [AHK02]. Indeed, it can only assert whether the set of all agents may or may not prevent the system from entering a failure state. It turns out that both the model-checking and the satisfiability problems are elementarily decidable and, precisely, 2ExpTime-complete [AHK02, Sch08].

Despite its powerful expressiveness, ATL$^\star$ suffers from a strong limitation, due to the fact that strategies are only treated 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) [CHP07], a logic that treats strategies in *two-player*

*turn-based games* as explicit *first-order objects*. In CHP-SL, the ATL$^\star$ formula $\langle\!\langle\{\alpha\}\rangle\!\rangle\psi$, for a system modeled by a CGS with agents $\alpha$ and $\beta$, becomes $\exists \mathtt{x}.\forall \mathtt{y}.\psi(\mathtt{x},\mathtt{y})$, i.e., "there exists a player-$\alpha$ strategy $\mathtt{x}$ such that for all player-$\beta$ strategies $\mathtt{y}$, the unique infinite path resulting from the two players following the strategies $\mathtt{x}$ and $\mathtt{y}$ satisfies the property $\psi$". The explicit treatment of strategies in this logic allows to state many properties not expressible in ATL$^\star$. In particular, it is shown in [CHP07] that ATL$^\star$, in the restricted case of two-agent turn-based games, corresponds to a proper one-alternation fragment of CHP-SL. The authors of that work have also shown that the model-checking problem for CHP-SL is decidable, although only a non-elementary algorithm for it, both in the size of system and formula, has been provided, leaving as an open question whether an algorithm with a better complexity exists or not. The complementary question about the decidability of the satisfiability problem for CHP-SL was also left open and it is not addressed in other papers apart the preliminary work [MMV10a].

While the basic idea exploited in [CHP07] to quantify over strategies and then to commit agents explicitly to certain of these strategies turns to be very powerful and useful [FKL10], CHP-SL still presents severe limitations. Among the others, it needs to be extended to the more general concurrent multi-agent setting. Also, the specific syntax considered there allows only a weak kind of strategy commitment. For example, CHP-SL does not allow different players to share the same strategy, suggesting that strategies have yet to become first-class objects in this logic. Moreover, an agent cannot change his strategy during a play without forcing the other to do the same.

These considerations, as well as all questions left open about decision problems, led us to introduce and investigate a new *Strategy Logic*, denoted SL, as a more general framework than CHP-SL, for explicit reasoning about strategies in *multi-agent concurrent games*. Syntactically, SL extends LTL by means of two *strategy quantifiers*, the existential $\langle\!\langle x\rangle\!\rangle$ and the universal $[\![x]\!]$, as well as *agent binding* $(a, x)$, where $a$ is an agent and $x$ a variable. Intuitively, these elements can be respectively read as *"there exists a strategy $x$"*, *"for all strategies $x$"*, and *"bind agent $a$ to the strategy associated with $x$"*. For example, in a CGS with three agents $\alpha$, $\beta$, $\gamma$, the previous ATL$^\star$ formula $\langle\!\langle\{\alpha,\beta\}\rangle\!\rangle\mathtt{G}\neg \mathit{fail}$ can be translated in the SL formula $\langle\!\langle\mathtt{x}\rangle\!\rangle\langle\!\langle\mathtt{y}\rangle\!\rangle[\![\mathtt{z}]\!](\alpha,\mathtt{x})(\beta,\mathtt{y})(\gamma,\mathtt{z})(\mathtt{G}\neg \mathit{fail})$. The variables $\mathtt{x}$ and $\mathtt{y}$ are used to select two strategies for the agents $\alpha$ and $\beta$, respectively, while $\mathtt{z}$ is used to select one for the agent $\gamma$ such that their composition, after the binding, results in a play where $\mathit{fail}$ is never met. Note that we can also require, by means of an appropriate choice of agent bindings, that agents $\alpha$ and $\beta$ share the same strategy, using the formula $\langle\!\langle\mathtt{x}\rangle\!\rangle[\![\mathtt{z}]\!](\alpha,\mathtt{x})(\beta,\mathtt{x})(\gamma,\mathtt{z})(\mathtt{G}\neg \mathit{fail})$. Furthermore, we may vary the structure of the game by changing the way the quantifiers alternate, as in the formula $\langle\!\langle\mathtt{x}\rangle\!\rangle[\![\mathtt{z}]\!]\langle\!\langle\mathtt{y}\rangle\!\rangle(\alpha,\mathtt{x})(\beta,\mathtt{y})(\alpha,\mathtt{z})(\mathtt{G}\neg \mathit{fail})$. In this case, $\mathtt{x}$ remains uniform w.r.t. $\mathtt{z}$, but $\mathtt{y}$ becomes dependent on it. Finally, we can change the strategy that one agent uses during the play without changing those of the other agents, by simply using nested bindings, as in the formula $\langle\!\langle\mathtt{x}\rangle\!\rangle\langle\!\langle\mathtt{y}\rangle\!\rangle[\![\mathtt{z}]\!]\langle\!\langle\mathtt{w}\rangle\!\rangle(\alpha,\mathtt{x})(\beta,\mathtt{y})(\gamma,\mathtt{z})(\mathtt{G}(\gamma,\mathtt{w})\mathtt{G}\neg \mathit{fail})$. The last examples intuitively show that SL is an extension of both ATL$^\star$ and CHP-SL. It is worth noting that the pattern of modal quantifications over strategies and binding to agents can be extended to other linear-time temporal logics than LTL, such as the linear $\mu$CALCULUS [Var88]. In fact, the use of LTL here is only a matter of simplicity in presenting our framework, and changing the embedded temporal logic only involves few side-changes in proofs and procedures.

## Introduction

As one of the main results in this thesis, we show that the model-checking problem is non-elementarily decidable. To gain this, we use an *automata-theoretic approach* [KVW00]. Precisely, we reduce the decision problem for our logic to the emptiness problem of a suitable *alternating parity tree automaton*, which is an *alternating tree automaton* (see [GTW02], for a survey) along with a *parity acceptance condition* [MS95]. Due to the operations of projection required by the elimination of quantifications on strategies, which induce at any step an exponential blow-up, the overall size of the required automaton is non-elementary in the size of the formula, while it is only polynomial in the size of the model. Thus, together with the complexity of the automata-nonemptiness calculation, we obtain that the model checking problem is in PTIME, w.r.t. the size of the model, and NONELEMENTARY, w.r.t. the size of the specification. Hence, the algorithm we propose is computationally not harder than the best one known for CHP-SL and, notably, a non-elementary data complexity improvement, (i.e., with respect to the size of the model). The latter is remarkable as in formal verification often the specification is very small (or even constant) with respect to the system. Moreover, we prove that our problem has a non-elementary lower bound. Specifically, it is $k$-EXPSPACE-HARD in the alternation number $k$ of quantifications in the specification. In addition to the model-checking problem for SL, we address the satisfiability problem, showing its undecidability [MMV10a]. The proof is based on the reduction from the *recurrent domino problem* proposed for the first time by Wang [Wan61]. A key role for this result is played by the unbounded model property of SL, which is proved in [MMPV12] and reported in this Thesis.

The contrast between the high complexity of the model-checking satisfiability problems for our logic and the elementary ones for ATL$^\star$ has spurred us to investigate syntactic fragments of SL, strictly subsuming ATL$^\star$, with a better complexity. In particular, by means of these sublogics, we want to understand why SL is computationally more difficult than ATL$^\star$. To this aim, we introduce *Nested-Goal*, *Boolean-Goal*, and *One-Goal Strategy Logic*, respectively denoted by SL[NG], SL[BG], and SL[1G]. These fragments encompass formulas in a special prenex normal form having nested temporal goals, Boolean combinations of goals, and a single goal at a time, respectively. For goal we mean an SL formula of the type $\flat\psi$, where $\flat$ is a binding prefix of the form $(\alpha_1, x_1), \ldots, (\alpha_n, x_n)$ containing all the involved agents and $\psi$ is an agent-full formula. With more detail, the idea behind SL[NG] is that, when in $\psi$ there is a quantification over a variable, then there are quantifications of all free variables contained in the inner subformulas. So, a subgoal of $\psi$ that has a variable quantified in $\psi$ itself cannot use other variables quantified out of this formula. Thus, goals can be only nested or combined with Boolean and temporal operators. SL[BG] and SL[1G] further restrict the use of goals. In particular, in SL[1G], each temporal formula $\psi$ is prefixed by a quantification-binding prefix $\wp\flat$ that quantifies over a tuple of strategies and binds them to all agents.

As main results about these fragments, we prove that the model-checking problem for SL[1G] is 2EXPTIME-COMPLETE, thus not harder than the one for ATL$^\star$. On the contrary, for SL[NG], it is NONELEMENTARY-HARD and thus we enforce the corresponding result for SL. Finally, by observing that SL[NG] includes SL[BG], we have that the model-checking problem for the latter is decidable with a 2EXPTIME lower-bound, given by SL[1G] model-checking complexity. The problem of determine its precise complexity is left open here.

For what the satisfiability regards, we prove that the problem remains undecidable for SL[BG], since the reduction provided for the entire SL already holds in this fragment. On the contrary, the problem is 2EXPTIME-COMPLETE for the smallest fragment SL[1G], thus not harder than ATL$^\star$.

To achieve all positive results about SL[1G], we introduce a fundamental property of the semantics of this logic, called *behavioral*[1], which allows us to strongly simplify the reasoning about strategies by reducing it to a set of reasonings about agent's choices (formally, agent's *actions*). This intrinsic characteristic of SL[1G], which unfortunately is not shared by the other fragments, asserts that, in a determined history of the play, the value of an existential quantified strategy depends only on the values of strategies, from which the first depends, on the same history. This means that, to choose an existential strategy, we do not need to know the entire structure of universal strategies, as for the whole SL, but only their values on the histories of interest. In other words, by means of the behavioral property, a one-shot second-order quantification over strategies can be eliminated in favor of a progressive first-order quantification over agent's choices. Technically, to describe the behavioral property, we make use of the machinery of *Skolem dependence function*, which defines a Skolemization procedure for SL, inspired by the one in first order logic (see [Hod01]).

By means of behavioral, we can modify the SL model-checking procedure via alternating tree automata in such a way that we avoid the projection operations by using a dedicated automaton that makes an action quantification for each node of the tree model. Consequently, the resulting automaton is only exponential in the size of the formula, independently from its alternation number. Thus, together with the complexity of the automata-nonemptiness calculation, we get that the model-checking procedure for SL[1G] is 2EXPTIME. As it is explained in Chapter 2, the behavioral property is fundamental to prove also the bounded-tree model property. Clearly, the behavioral property also holds for ATL$^\star$, as it is included in SL[1G]. In particular, although it has not been explicitly stated, this property is crucial for most of the results achieved in literature about ATL$^\star$ by means of automata (see [Sch08], as an example). Moreover, we believe that our proof techniques are of independent interest and applicable to other logics as well (see for example [MP14]).

The formal representation in SL can be exploited to address several problems. For instance, we can investigate the problem of *Rational Synthesis*. *Synthesis* is the automated construction of a system from its specification. The basic idea is simple and appealing: instead of developing a system and verifying that it adheres to its specification, we would like to have an automated procedure that, given a specification, constructs a system that is correct by construction. The first formulation of synthesis goes back to Church [Chu63]; the modern approach to synthesis was initiated by Pnueli and Rosner, who introduced LTL (linear temporal logic) synthesis [PR89]. The LTL *synthesis problem* receives as input a specification in LTL and outputs a reactive system modeled by a finite-state transducer satisfying the given specification — if such exists. It is important to distinguish between input signals, assigned by the environment, and output signals, assigned by the system. A system should be able to cope with all values of the input signals, while setting the output signals to desired values [PR89].

[1] We use this term as it has a direct correspondence with the "behavioral" concept used in game theory [Mye97, MMS13, MMS14].

Therefore, the quantification structure on input and output signals is different. Input signals are universally quantified while output signals are existentially quantified.

Modern systems often interact with other systems. For example, the clients interacting with a server are by themselves distinct entities (which we call agents) and are many times implemented by systems. In the traditional approach to synthesis, the way in which the environment is composed of its underlying agents is abstracted. In particular, the agents can be seen as if their only objective is to conspire to fail the system. Hence the term "hostile environment" that is traditionally used in the context of synthesis. In real life, however, many times agents have goals of their own, other than to fail the system. The approach taken in the field of algorithmic game theory [NRTV07] is to assume that agents interacting with a computational system are *rational*, i.e., agents act to achieve their own goals. Assuming agents rationality is a restriction on the agents behavior and is therefore equivalent to softening the universal quantification on the environment. [2] Thus, the following question arises: can system synthesizers capitalize on the rationality and goals of agents interacting with the system?

In [FKL10], Fisman et al. positively answered this question by introducing and studying *rational synthesis*. The input to the rational-synthesis problem consists of LTL formulas specifying the objectives of the system and the agents that constitute the environment, and a solution concept, e.g., dominant strategies, Nash Equilibria, and the like. The atomic propositions over which the objectives are defined are partitioned among the system and the agents, so that each of them controls a subset of the propositions. The desired output is a strategy profile such that the objective of the system is satisfied in the computation that is the outcome of the profile, and the agents that constitute the environment have no incentive to deviate from the strategies suggested to them (formally, the profile is an equilibrium with respect to the solution concept). Fisman et al. showed that there are specifications that cannot be realized in a hostile environment but are realizable in a rational environment. Moreover, the rational-synthesis problem for LTL and common solution concepts used in game theory can be solved in 2EXPTIME, thus it complexity coincides with that of usual synthesis.

In this thesis, we present the following three contributions on Rational Synthesis, as they have been introduced in [KPV14]. First, we suggest an alternative definition to rational synthesis, in which the agents are rational but not cooperative. Second, we study a richer specification formalism, where the objectives of the system and the agents are not Boolean but quantitative. Third, we show that all these variants of the rational synthesis problems can be reduced to the model checking in fragments of *Strategy Logic* [MMPV14]. Before we describe our contributions in more detail, let us highlight a different way to consider rational synthesis and our contribution here. *Mechanism design* is a field in game theory and economics studying the design of games whose outcome (assuming agents rationality) achieves some goal [NR01, NRTV07]. The outcome of traditional games depends on the final position of the game. In contrast, the systems we reason about maintain an *on-going interaction* with their environment, and we reason about their behavior by referring not to their final state (in fact, we consider non-terminating systems, with no final state) but rather

[2] Early work on synthesis has realized that the universal quantification on the behaviors of the environment is often too restrictive. The way to address this point, however, has been by adding assumptions on the environment, which can be part of the specification (cf., [CHJ08]).

to the *language* of computations that they generate. Rational synthesis can be viewed as a variant of mechanism design in which the game is induced by the objective of the system, and the objectives of both the system and the agents refer to their on-going interaction and are specified by temporal-logic formulas. Our contributions here correspond to the classic setting assumed in mechanism design: the agents need not be cooperative, and the outcome is not Boolean.

We argue that the definition of rational synthesis in [FKL10] is *cooperative*, in the sense that the agents that constitute the environment are assumed to follow the strategy profile suggested to them (as long as it is in an equilibrium). Here, we consider also a *non-cooperative* setting, in which the agents that constitute the environment may follow any strategy profile that is in an equilibrium, and not necessarily the one suggested to them by the synthesis algorithm. In many scenarios, the cooperative setting is indeed too optimistic, as the system cannot assume that the environment, even if it is rational, would follow a suggested strategy, rather than a strategy that is as good for it. Moreover, sometimes there is no way to communicate with the environment and suggest a strategy for it. From a technical point of view, we show that the non-cooperative setting requires reasoning about *all* possible equilibria, yet, despite this more sophisticated reasoning, it stays 2EXPTIME-COMPLETE. We achieve the upper bound by reducing rational synthesis to the model-checking problem for SL. While the model-checking problem for strategy logic is in general non-elementary, we show that it is possible to express rational synthesis in SL[NG, 1-ALT] [3], which leads to the desired complexity. It is important to observe the following difference between the cooperative and the non-cooperative settings. In the cooperative one, we synthesize strategies for all agents, with the assumption that the agent that corresponds to the system always follows his suggested strategy and the agents that constitute the environment decide in a rational manner whether to follow their strategies. On the other hand, in the non-cooperative setting, we synthesize a strategy only for the agent that corresponds to the system, and we assume that the agents that constitute the environment are rational, thus the suggested strategy has to win against all rational behaviors of the environment.

Our second contribution addresses a weakness of the classical synthesis problem, a weakness that is more apparent in the rational setting. In classical synthesis, the specification is Boolean and describes the expected behavior of the system. In many applications, systems can satisfy their specifications at different levels of quality. Thus, synthesis with respect to Boolean specifications does not address designers needs. This latter problem is a real obstacle, as designers would be willing to give up manual design only after being convinced that the automatic procedure that replaces it generates systems of comparable quality. In the last years we see a lot of effort on developing formalisms that would enable the specification of such quality measures [BCHJ09, ABK13].

Classical applications of game theory consider games with quantitative payoffs. In the Boolean setting, we assumed that the payoff of an agent is, say, 1, if its objective is satisfied, and is 0 otherwise. In particular, this means that agents whose objectives are not satisfied have no incentive to follow any strategy, even if the profile satisfies the solution concept. In real-life, rational objectives are rarely Boolean. Thus, even beyond our goal of synthesizing

[3] By SL$[k$-ALT$]$ we denote the fragment of SL in which only a $k$ number of alternation is allowed.

systems of high quality, the extension of the synthesis problem to the rational setting calls also for an extension to a quantitative setting. Unfortunately, the full quantitative setting is undecidable already in the context of model checking [Hen10]. In [FKL10], Fisman et al. extended cooperative rational synthesis to objectives in the multi-valued logic LLTL, where specifications take truth values from a finite lattice.

We introduce here a new quantitative specification formalism, termed *Objective* LTL , (OLTL, for short). We first define the logic, and then study its rational synthesis. Essentially, an OLTLspecification is a pair $\theta = \langle \Psi, \mathsf{f} \rangle$, where $\Psi = \langle \psi_1, \psi_2, \ldots, \psi_m \rangle$ is a tuple of LTL formulas and $\mathsf{f} : \{0,1\}^m \to \mathbb{Z}$ is a *reward function*, mapping Boolean vectors of length $m$ to an integer. A computation $\pi$ then maps $\theta$ to a reward in the expected way, according to the subset of formulas that are satisfied in $\pi$. In the rational synthesis problem for OLTL, the input consists of OLTLspecifications for the system and the other agents, and the goal of the system is to maximize its reward with respect to environments that are in an equilibrium. Again, we distinguish between a cooperative and a non-cooperative setting. Note that the notion of an equilibrium in the quantitative setting is much more interesting, as it means that all agents in the environment cannot expect to increase their payoffs. We show that the quantitative setting is not more complex than the non-quantitative one, thus quantitative rational synthesis is complete for 2EXPTIME in both the cooperative and non-cooperative settings.

We conclude this thesis by presenting a result that aims to combine strategic specifications with pushdown systems. Starting from the works on module checking, two significant directions have been taken in open-system verification. One concerns extending the framework to more sophisticated systems while maintaining the dichotomy system-environment states in modeling. In this context, worthy of mention is the work on *pushdown module checking* [BMP10]. This has the merit of having handled the verification of infinite-state open systems and, thanks to the fact that the infinite number of states is induced by a recursive structure of finite size, the problem turns out to be decidable and precisely 3EXPTIME-COMPLETE for specifications in CTL$^\star$. Another direction has instead completely redesigned the module checking approach in order to handle the more involved scenario of multi-agent (concurrent) systems. This is, for instance, the above mentioned case of ATL$^\star$. However, the two approaches are incomparable as in module checking it is possible to use nondeterministic strategies.

Despite the undoubted utility of considering, from one hand, infinite-state open-system models induced by finite-size recursive structures and, from the other hand, multi-agent specifications, to the best of our knowledge no work has been devoted to the combination of the two.

In this thesis, we consider multi-agent pushdown systems and address the related model checking problem for specifications expressed in ATL$^\star$. To this aim, we first introduce *pushdown game structures* to properly model the infinite-state multi-agent system and formalize the model checking question. Then, by means of an automata-theoretic approach, we provide a 3EXPTIME solution to the addressed problem. Precisely, we construct a doubly-exponential size *pushdown parity tree automaton* that collects all execution trees satisfying the ATL$^\star$ formula. Then by using the fact that the emptiness of this automaton can be checked in exponential time [KPV02], we get the desired result. We also provide a 2EXPSPACE lower-bound

by showing a reduction from the model checking problem for one-counter systems against quantitative ATL$^\star$ specifications [Ves14].

**Related works.** We now discuss some work related to the results included in the thesis.

**Logics for strategic reasoning representation** Several works have focused on extensions of ATL and ATL$^\star$ to incorporate more powerful strategic constructs. Among them, we recall *Alternating-Time* $\mu$CALCULUS (A$\mu$CALCULUS, for short) [AHK02], *Game Logic* (GL, for short) [AHK02], *Quantified Decision Modality* $\mu$CALCULUS (QD$\mu$CALCULUS, for short) [Pin07], *Coordination Logic* (CL, for short) [FS10], (ATL *with plausibility* (ATL$^+$, for short) [BJD08], (ATL *with Irrevocable strategies* (IATL, for short) [ÅGJ07], (*Memoryful* ATL$^\star$ (mATL$^\star$, for short) [MMV10b], *Basic Strategy-Interaction Logic* (BSIL, for short) [WHY11] *Temporal Cooperation Logic* (TCL, for short) [HSW13], *Alternating-time Temporal Logic with Explicit Actions* (ATLEA, for short) [HLW13] and some extensions of ATL$^\star$ considered in [BLLM09]. A$\mu$CALCULUS and QD$\mu$CALCULUS are intrinsically different from SL (as well as from CHP-SL and ATL$^\star$) as they are obtained by extending the propositional $\mu$-calculus [Koz83] with strategic modalities. CL is similar to QD$\mu$CALCULUS but with LTL temporal operators instead of explicit fixpoint constructors. GL is strictly included in CHP-SL, in the case of two-player turn-based games, but it does not use any explicit treatment of strategies, neither it does the extensions of ATL$^\star$ introduced in [BLLM09], which consider restrictions on the memory for strategy quantifiers. ATL$^+$ enables to express rationality assumptions of intelligent agents in ATL. In IATL, the semantics of the logic ATL is changed in a way that, in the evaluation of the goal, agents can be forced to keep the strategy they have chosen in the past in order to reach the state where a goal is evaluated. mATL$^\star$ enriches ATL$^\star$ by giving the ability to agents to "relent" and change their goals and strategies depending on the history of the play. BSIL allows to specify behaviors of a system that can cooperate with several strategies of the environment for different requirements. TCL extends ATL by allowing successive definitions of agent strategies, with the aim of using the collaborative power of groups of agents to enforce different temporal objectives. ATLEA introduces explicit actions in the logic ATL to reason about abilities of agents under commitments to play precise actions. Thus, all above logics are different from SL, which we recall it aims to be a minimal but powerful logic to reason about strategic behavior in multi-agent systems.

At roughly the same time we have conceived Strategy Logic, another generalization of ATL$^\star$, named ATL$^\star$ *with Strategy Contexts*, which results to be very expressive but a proper sublogic of SL, has been considered in [DLM10] (see also [DLM12, LM13] for more recent works). In this logic, a quantification over strategies does not reset the strategies previously quantified but allows to maintain them in a particular context in order to be reused. This makes the logic much more expressive than ATL$^\star$. On the other hand, as it does not allow agents to share the same strategy, it is not comparable with the fragments we have considered in this paper.

Recently, several extensions of SL have been also investigated. *Updating Strategy Logic* (U-SL, for short) has been considered in [CBC13] where, in addition to SL, an agent can

refine its own strategies by means of an "unbinder" operator, which explicitly deletes the binding of a strategy to an agent. In [Bel14], an epistemic extension of SL with modal operators for individual knowledge has been considered, showing that the complexity of model checking for this logic is not worse than the one for (non-epistemic) SL. Last but not least, in [ČLMM14] a BDD-based model checker for the verification of systems against specifications expressed in the epistemic extension of SL, named MCMAS-SLK, has been introduced.

**Synthesis** The understanding that synthesis corresponds to a game in which the objective of each player is to satisfy his specification calls for a mutual adoption of ideas between formal methods and game theory. In *rational synthesis*, introduced in [FKL10], synthesis is defined in a way that takes into an account the rationality of the agents that constitute the environment and involves and assumption that an agent cooperates with a strategy in which his objective is satisfied. Here, we extend the idea and consider also *non-cooperative* rational synthesis, in which agents need not cooperate with suggested strategies and may prefer different strategies that are at least as beneficial for them.

Many variants of the classical synthesis problem has been studied. It is interesting to examine the combination of the rational setting with the different variants. To start, the cooperative and non-cooperative settings can be combined into a framework in which one team of agents is competing with another team of agents, where each team is internally cooperative, but the two teams are non-cooperative. Furthermore, we plan to study rational synthesis with *incomplete information* [KV99], where agents can view only a subset of the signals that other agents output, and rational *stochastic* synthesis [CMJ04], which models the unpredictability of nature and involves stochastic agents that assign values to their output signals by means of a distribution function. Beyond a formulation of the richer settings, one needs a corresponding extension of strategy logic and its decision problems.

As discussed above, classical applications of game theory consider games with quantitative payoffs. We added a quantitative layer to LTL by introducing Objective-LTL and studying its rational synthesis. In recent years, researchers have developed more refined quantitative temporal logics, which enable a formal reasoning about the quality of systems. In particular, we plan to study rational synthesis for the multi-valued logics LTL[F] [ABK13], which enables a prioritization of different satisfaction possibilities, and LTL[D] [ABK14], in which discounting is used in order to reduce the satisfaction value of specifications whose eventualities are delayed. The rational synthesis problem for these logics induce a game with much richer, possibly infinitely many, profiles, making the search for a stable solution much more challenging.

**Pushdown systems** In recent years, model checking of pushdown systems has received a lot of attention, largely due to the ability of these systems to capture the flow of procedure of calls and returns in programs [ABE$^+$05]. The work in this area started with Muller and Schupp, who showed that the monadic second-order theory of graphs induced by pushdown systems is decidable [MS85]. Walukiewicz in [Wal96] showed that the model checking for pushdown systems with respect to *modal $\mu$-calculus* is EXPTIME-COMPLETE. The problem remains

EXPTIME-COMPLETE also for CTL and LTL, while it becomes 2EXPTIME-COMPLETE for CTL$^\star$ [Wal00, BEM97]. In [BMP10], open pushdown systems along with the module checking paradigm have been considered. This setting has been investigated under several restrictions including the imperfect-information case [ALM$^+$13].

CHAPTER 1

# Strategy Logic

This preliminary chapter is organized as follows. In Section 1.1 we introduce the syntax and the semantics of *Strategy Logic*, together with the auxiliary definition of the underlying framework, namely *concurrent game structure*. Then, we discuss its expressiveness by showing examples, most of them borrowed by come classic problem in Game Theory. In Section 1.2 we discuss the model-checking problem for this logic, by showing a reduction from the satisfiability problem for QPTL, which is known to be NONELEMENTARY-COMPLETE. Finally, in Section 1.3 we show that the satisfiability problem for SL is undecidable, by means of a reduction from the *recurrent domino problem*, introduced in [Wan61].

## 1.1 Syntax and Semantics

*Strategy Logic* [MMV10a] (SL, for short) is an extension of the classic linear-time temporal logic LTL [Pnu77] along with the concepts of strategy quantifications and agent binding, which formalism allows to express strategic plans over temporal goals. The main distinctive feature of this formalism *w.r.t.* other logics with the same aim resides in the decoupling strategy instantiations, done through the quantifications, from their applications, by means of bindings. Consequently, the logic is not simply propositional but predicative, since we treat strategies as a first order concept via the use of agents and variables as explicit syntactic elements. This fact let us to write Boolean combinations and nesting of complex predicates, representing each one a different temporal goal, linked together by some common strategic choices.

### 1.1.1 Underlying framework

As semantic framework for SL, we use the *graph-based model* for *multi-player games* named *concurrent game structure* [AHK02], which is a generalization of *Kripke structures* [Kri63] and *labeled transition systems* [Kel76]. It allows to model *multi-agent systems* viewed as extensive form games, in which players perform *concurrent actions* to trigger different transitions over the graph.

**Definition 1.1.1 (Concurrent Game Structures)** *A* concurrent game structure *(*CGS*, for short) is a tuple* $\mathcal{G} \triangleq \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}, \mathrm{St}, \mathsf{tr}, \mathsf{ap}, s_{\mathrm{o}} \rangle$*, 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*,* $s_{\mathrm{o}} \in \mathrm{St}$ *is a designated* initial state*, and* $\mathsf{ap} : \mathrm{St} \to 2^{\mathrm{AP}}$ *is a* labeling function *that maps each state to the set of atomic propositions true in that state. Let* $\mathrm{Dc} \triangleq \mathrm{Ac}^{\mathrm{Ag}}$ *be the set of* decisions*,* i.e.*, functions from* Ag *to* Ac *representing the choices of an action for each*

*agent.* [1] *Then,* $\mathsf{tr} : \mathrm{St} \times \mathrm{Dc} \to \mathrm{St}$ *is a* transition function *mapping a pair of a state and a decision to a state.*

To get familiar with the concept of CGS, we present here some running example of simple concurrent games.

We start by modeling the *paper, rock, and scissor* game.

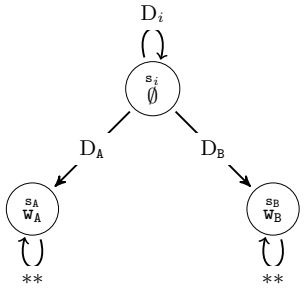



Figure 1.1: The CGS $\mathcal{G}_{PRS}$.

**Example 1.1.1 (Paper, Rock, and Scissor)** *Consider the classic two-player concurrent game* paper, rock, and scissor *(PRS, for short) as represented in Figure 1.1, where a play continues until one of the participants catches the move of the other. Vertexes are states of the game and labels on edges represent decisions of agents or sets of them, where the symbol $*$ is used in place of every possible action. In this specific case, since there are only two agents, the pair of symbols $**$ indicates the whole set* $\mathrm{Dc}$ *of decisions. The agents "Alice" and "Bob" in* $\mathrm{Ag} \triangleq \{\mathtt{A}, \mathtt{B}\}$ *have as possible actions those in the set* $\mathrm{Ac} \triangleq \{\mathtt{P}, \mathtt{R}, \mathtt{S}\}$*, which stand for "paper", "rock", and "scissor", respectively. During the play, the game can stay in one of the three states in* $\mathrm{St} \triangleq \{\mathsf{s}_i, \mathsf{s}_\mathtt{A}, \mathsf{s}_\mathtt{B}\}$*, which represent, respectively, the waiting moment, named* idle*, and the two winner positions. The latter ones are labeled accordingly with one of the atomic propositions in* $\mathrm{AP} \triangleq \{\mathtt{w}_\mathtt{A}, \mathtt{w}_\mathtt{B}\}$*, in order to represent who is the winner between* A *and* B*. The catch of one action over another is described by the relation* $C \triangleq \{(\mathtt{P}, \mathtt{R}), (\mathtt{R}, \mathtt{S}), (\mathtt{S}, \mathtt{P})\} \subseteq \mathrm{Ac} \times \mathrm{Ac}$*. We can now define the* CGS $\mathcal{G}_{PRS} \triangleq \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}, \mathrm{St}, \mathsf{tr}, \mathsf{ap}, \mathsf{s}_i \rangle$ *for the PRS game. The labeling is given by* $\mathsf{L}(\mathsf{s}_i) \triangleq \emptyset$*,* $\mathsf{L}(\mathsf{s}_\mathtt{A}) \triangleq \{\mathtt{w}_\mathtt{A}\}$*, and* $\mathsf{L}(\mathsf{s}_\mathtt{B}) \triangleq \{\mathtt{w}_\mathtt{B}\}$*. Moreover, the transition function is defined as follows, where* $\mathrm{D}_\mathtt{A} \triangleq \{\mathsf{dc} \in \mathrm{Dc}_{\mathcal{G}_{PRS}} : (\mathsf{dc}(\mathtt{A}), \mathsf{dc}(\mathtt{B})) \in C\}$ *and* $\mathrm{D}_\mathtt{B} \triangleq \{\mathsf{dc} \in \mathrm{Dc}_{\mathcal{G}_{PRS}} : (\mathsf{dc}(\mathtt{B}), \mathsf{dc}(\mathtt{A})) \in C\}$ *are the sets of winning decisions for the two agents: if* $s = \mathsf{s}_i$ *and* $\mathsf{dc} \in \mathrm{D}_\mathtt{A}$ *then* $\mathsf{tr}(s, \mathsf{dc}) \triangleq \mathsf{s}_\mathtt{A}$*, else if* $s = \mathsf{s}_i$ *and* $\mathsf{dc} \in \mathrm{D}_\mathtt{B}$ *then* $\mathsf{tr}(s, \mathsf{dc}) \triangleq \mathsf{s}_\mathtt{B}$*, otherwise* $\mathsf{tr}(s, \mathsf{dc}) \triangleq s$*. Note that, when none of the two agents catches the action of the other, i.e., the used decision is in* $\mathrm{D}_i \triangleq \mathrm{Dc}_{\mathcal{G}_{PRS}} \setminus (\mathrm{D}_\mathtt{A} \cup \mathrm{D}_\mathtt{B})$*, the play remains in the idle state to allow another try, otherwise it is stuck in a winning position forever.*

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

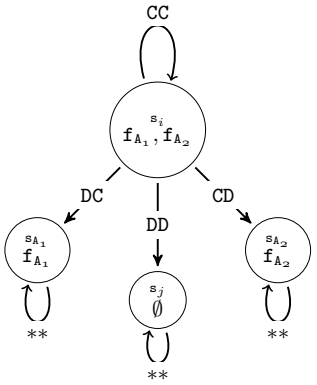



Figure 1.2: The CGS $\mathcal{G}_{PD}$.

**Example 1.1.2 (Prisoner's Dilemma)** *In the* prisoner's dilemma *(PD, for short), two accomplices are interrogated in separated rooms by the police, which offers them the same agreement. If one defects, i.e., testifies for the prosecution against the other, while the other cooperates, i.e., remains silent, the defector goes free and the silent accomplice goes to jail. If both cooperate, they remain free, but will be surely interrogated in the next future waiting*

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

*for a defection. On the other hand, if they both defect, both go to jail. Note that no one of the two prisoners knows about the choice made by the other. This tricky situation can be modeled by the* CGS $\mathcal{G}_{PD} \triangleq \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}, \mathrm{St}, \mathsf{tr}, \mathsf{ap}, \mathsf{s}_i \rangle$ *depicted in Figure 1.2 on the preceding page, where the agents "Accomplice-1" and "Accomplice-2" in* $\mathrm{Ag} \triangleq \{\mathtt{A}_1, \mathtt{A}_2\}$ *can chose an action in* $\mathrm{Ac} \triangleq \{\mathtt{C}, \mathtt{D}\}$*, which stand for "cooperation" and "defection", respectively. There are four states in* $\mathrm{St} \triangleq \{\mathsf{s}_i, \mathsf{s}_{\mathtt{A}_1}, \mathsf{s}_{\mathtt{A}_2}, \mathsf{s}_j\}$*. In the idle state* $\mathsf{s}_i$ *the agents are waiting for the interrogation, while* $\mathsf{s}_j$ *represents the jail for both of them. Instead the remaining states* $\mathsf{s}_{\mathtt{A}_1}$ *and* $\mathsf{s}_{\mathtt{A}_2}$ *indicate the situations in which only one of the agents becomes definitely free. To characterize the different meaning of these states, we use the atomic propositions in* $\mathrm{AP} \triangleq \{\mathsf{f}_{\mathtt{A}_1}, \mathsf{f}_{\mathtt{A}_2}\}$*, which denote who is "free", by defining the following labeling:* $\mathsf{L}(\mathsf{s}_i) \triangleq \{\mathsf{f}_{\mathtt{A}_1}, \mathsf{f}_{\mathtt{A}_2}\}$, $\mathsf{L}(\mathsf{s}_{\mathtt{A}_1}) \triangleq \{\mathsf{f}_{\mathtt{A}_1}\}$, $\mathsf{L}(\mathsf{s}_{\mathtt{A}_2}) \triangleq \{\mathsf{f}_{\mathtt{A}_2}\}$*, and* $\mathsf{L}(\mathsf{s}_j) \triangleq \emptyset$*. The transition function* $\mathsf{tr}$ *can be easily deduced by the figure.*

We can analyze also an extended version of the well-known prisoner's dilemma [OR94] in which also the actions of the police are taken into account.

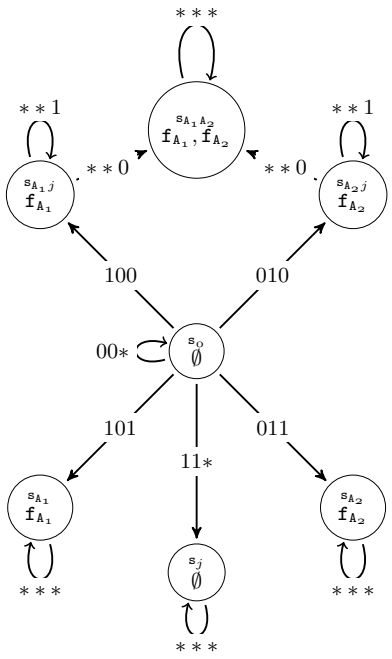



Figure 1.3: The CGS $\mathcal{G}_{PPD}$.

**Example 1.1.3 (Prisoners and Police's Dilemma)** *In the* prisoner and police's dilemma *(PPD, for short), apart from the classic agreement of the PD, the two accomplices also know that they can try to gain a better sentence by the judge if one spontaneously defects without being interrogated by police, since he is considered a "good willing man". In this case, indeed, if the other cooperates, the defector becomes definitely free, while the other goes to jail with the possibility to eventually be released. It is important, however, that no both of them defect, otherwise the police can subtle act as they were interrogated. Moreover, differently from the PD, they are not free during the time in which they can be interrogated. This complex situation, can be modeled by the* CGS $\mathcal{G}_{PPD} \triangleq \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}, \mathrm{St}, \mathsf{tr}, \mathsf{ap}, s_0 \rangle$ *depicted in Figure 1.3, where there are three agents in* $\mathrm{Ag} \triangleq \{\mathtt{A}_1, \mathtt{A}_2, \mathtt{P}\}$*, with* $\mathtt{P}$ *being the police, and all of them can execute the two abstract actions in* $\mathrm{Ac} \triangleq \{0, 1\}$*. For the accomplices,* $0$ *and* $1$ *have the meaning of "cooperate" and "defect". For the police, on the contrary, they mean "wait" and "interrogate" in all the states but those in which one of the accomplices can eventually be released, where the meaning is "release" and "maintain", instead. The set of states for the game is given by* $\{\mathsf{s}_i, \mathsf{s}_{\mathtt{A}_1}, \mathsf{s}_{\mathtt{A}_2}, \mathsf{s}_j, \mathsf{s}_{\mathtt{A}_1 j}, \mathsf{s}_{\mathtt{A}_2 j}, \mathsf{s}_{\mathtt{A}_1\mathtt{A}_2}\}$*. The idle state* $\mathsf{s}_i$ *the two prisoners are waiting to be interrogated by police. They can even decide to defect before interrogation. The states* $\mathsf{s}_{\mathtt{A}_1}$ *and* $\mathsf{s}_{\mathtt{A}_2}$ *denotes the situation in which only one prisoner becomes definitely free. Moreover, the states* $\mathsf{s}_{\mathtt{A}_1 j}$ *and* $\mathsf{s}_{\mathtt{A}_2 j}$ *indicate when one of the prisoner is free while the other in the jail is waiting for his release. Finally,* $\mathsf{s}_{\mathtt{A}_1\mathtt{A}_2}$ *denotes the state in which both have gained definitely the freedom. To represent the different meaning of these states, we use the atomic propositions* $\mathsf{f}_{\mathtt{A}_i}$ *to denote that the prisoner* $\mathtt{A}_i$ *is free. Both the labeling function* $\mathsf{ap}$ *and the transition*

*function* tr *can be extracted from the figure.*

In addition to PPD, we model a very simple *preemptive scheduling* protocol for the access of processes to a shared resource.

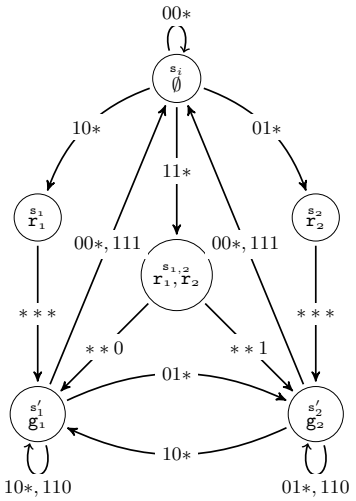



Figure 1.4: The CGS $\mathcal{G}_{PS}$.

**Example 1.1.4 (Preemptive Scheduling)** *Consider the following* preemptive scheduling *protocol (PS, for short) describing the access rules of two processes to a shared resource in a preemptive way. When the resource is free and only one process asks for it, this process receives directly the grant. Instead, if there is a competition of requests, it is the scheduler that, in a nondeterministic way, determines who can access to the resource. Finally, in case one process owns the resource while the other asks for it, the scheduler can choose to make or not a preemption. These rules are formalized in the* CGS $\mathcal{G}_{PS} \triangleq \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}, \mathrm{St}, \mathtt{tr}, \mathtt{ap}, s_0 \rangle$ *of Figure 1.4, where the agents "Process-1", "Process-2" and "Scheduler" in* $\mathrm{Ag} \triangleq \{\mathtt{P}_1, \mathtt{P}_2, \mathtt{S}\}$ *can choose between the two abstract actions in* $\mathrm{Ac} \triangleq \{0, 1\}$. *The processes use the actions* 0 *and* 1 *to send or not the request to the scheduler, while this uses them in order to decide who can have the access to the resource in a situation of competition. There are five states* $\mathrm{St} \triangleq \{\mathtt{s}_i, \mathtt{s}_1, \mathtt{s}_2, \mathtt{s}_{1,2}, \mathtt{s}'_1, \mathtt{s}'_2\}$ *in which the protocol can reside: the idle state* $\mathtt{s}_i$ *in which the resource is free; the three states* $\mathtt{s}_1$, $\mathtt{s}_2$, *and* $\mathtt{s}_{1,2}$ *in which* $\mathtt{P}_1$, $\mathtt{P}_2$ *or both are requesting the resource; the two states* $\mathtt{s}'_1$ *and* $\mathtt{s}'_2$ *in which the resource has been finally granted to* $\mathtt{P}_1$ *and* $\mathtt{P}_2$, *respectively. To represent all information associated, we use the atomic propositions in* $\mathrm{AP} \triangleq \{\mathtt{r}_1, \mathtt{r}_2, \mathtt{g}_1, \mathtt{g}_2\}$, *where* $\mathtt{r}_i$ *represents the request of* $\mathtt{P}_i$, *while* $\mathtt{g}_i$ *the fact that the resource has been granted to* $\mathtt{P}_i$.

As last example in which it is possible to discuss about the ability of an agent to be safe with respect to the behavior of malicious agents, we consider a generalization of the well known robber-vs-cops game (see [AF84] for a survey).

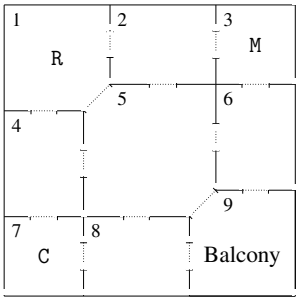



Figure 1.5: Romeo and Juliet game.

**Example 1.1.5 (Romeo and Juliet Game)** *In the* Romeo and Juliet *game, Romeo, being able to move inside Juliet's house, aims to reach her balcony, trying to avoid meeting both the Montagues and Capulets families, who are able to move concurrently with him in the house. Graphically, the house is divided in rooms and each room has doors on all internal walls. The topology of the house depends on the status of these doors that can be closed or open.*

*Additionally, Romeo is helped by the accomplice Shakespeare that controls the doors of the rooms. Specifically, at each step of the game, Shakespeare can switch the status of at most two doors at a time (i.e., from open to closed or viceversa). Hence, he can modify the shape of the house during a play. We consider Romeo meeting the families in two cases: if they are in the same room in a certain phase of the play (Romeo is captured) or if they are*

*in adjacent rooms with the connecting door open (Romeo is shot). In each step, the agents Romeo, Montagues family, and Capulets family can independently decide to stay in the room or, in the case a connection door to another room is open, move to this one. A sample game given in Figure 1.5 on the facing page is made by nine rooms, with Room 9 being the balcony, Romeo in Room 1, Montagues family in Room 3 and Capulets family in Room 7, respectively, and all the doors open as initial state of the game. We denote with* $\mathrm{R} = [1, 9]$ *the set of rooms and* $\mathrm{D} = \{d = (r_1, r_2) \in \mathrm{R} \times \mathrm{R} : r_1 \text{ is adjacent to } r_2\}$ *the set of doors.*

*We can model this situation with a* CGS $\mathcal{G}_{RJ} = \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}, \mathrm{St}, \mathtt{tr}, \mathtt{ap}, s_0 \rangle$ *with* $\mathrm{AP} \triangleq \{\mathtt{c}, \mathtt{r}\}$ *standing for* Captured *or* Shot *(*`c`*) and* Reached *(*`r`*);* $\mathrm{Ag} \triangleq \{\mathtt{R}, \mathtt{M}, \mathtt{C}, \mathtt{S}\}$*, standing for* Romeo *(*`R`*),* Montagues *(*`M`*),* Capulets *(*`C`*),* Shakespeare *(*`S`*), respectively;* $\mathrm{Ac} \triangleq \mathrm{M} \cup \mathrm{C}$ *where* $\mathrm{M} \triangleq \{\mathtt{stay}, \mathtt{up}, \mathtt{down}, \mathtt{left}, \mathtt{right}\}$ *is the set of moves for Romeo and Families and* $\mathrm{C} = \{d \in 2^{\mathrm{D}} : |d| \leq 2\}$ *be the set of possible switching of doors for Shakespeare, in which the empty set stands for no switching;* $\mathrm{St} \triangleq \mathrm{R} \times \mathrm{R}^2 \times \{0, 1\}^{\mathrm{D}}$ *be the set of states in which the first coordinate stands for the position of Romeo, the second and third coordinates stand for the positions of Montagues and Capulets, respectively, and the fourth coordinate is a function describing the status open or closed of each door in the house. The initial state is given by* $(1, 3, 7, \mathtt{f}_0)$ *where* $\mathtt{f}_0 : \mathrm{D} \to \{0, 1\}$ *is the function constant to* $0$*, i.e., all the doors are open. The labeling function is given as follows: for each state* $s = (n, m_1, m_2, \mathtt{f}) \in \mathrm{St}$*,* $\mathtt{r} \in \mathtt{L}(s)$ *iff* $n = 9$*,*[2] $\mathtt{c} \in \mathtt{L}(s)$ *iff there exists* $i \in \{1, 2\}$ *such that either* $n = m_i$ *or* $(n, m_i) \in \mathrm{D}$ *and* $\mathtt{f}((n, m_i)) = 0$*.*[3] *The transition function can be easily deduced by the figure. For the sake of simplicity, all the states labeled with* `r` *only have a loop in the game.* [4]

### 1.1.2 Syntax

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

**Definition 1.1.2 (**SL **Syntax)** SL formulas *are built inductively from the sets of atomic propositions* AP*, variables* Vr*, and agents* Ag*, by using the following grammar, where* $p \in \mathrm{AP}$*,* $x \in \mathrm{Vr}$*, and* $a \in \mathrm{Ag}$*:*

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

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

Observe that, by construction, LTL is a proper syntactic fragment of SL, *i.e.*, LTL $\subset$ SL. In order to abbreviate the writing of formulas, we use the boolean values true `t` and false `f` and

[2] Romeo has reached the balcony.

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

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

the well-known temporal operators future $\mathtt{F}\varphi \triangleq \mathtt{t}\ \mathtt{U}\varphi$ and globally $\mathtt{G}\varphi \triangleq \mathtt{f}\ \mathtt{R}\varphi$. Moreover, we use the italic letters $x, y, z, \ldots$, possibly with indexes, as meta-variables on the variables $\mathtt{x}, \mathtt{y}, \mathtt{z}, \ldots$ in Vr.

A first classic notion related to the syntax of SL is that of *subformula*, *i.e.*, a syntactic expression that is part of an a priori given formula. By $\mathsf{sub}(\varphi)$ we formally denote the set of subformulas of an SL formula $\varphi$. For instance, consider $\varphi = \langle\!\langle \mathtt{x} \rangle\!\rangle (\alpha, \mathtt{x})(\mathtt{Fp})$. Then, it is immediate to see that $\mathsf{sub}(\varphi) = \{\varphi, (\alpha, \mathtt{x})(\mathtt{Fp}), (\mathtt{Fp}), \mathtt{p}, \mathtt{t}\}$.

Usually, in predicative logics, we need the concepts of *free* and *bounded* placeholders, to correctly define the meaning of a formula. In SL, we have two different kind of placeholders: variables and agents. The former is used in the strategy quantifications, the latter to commit an agent, by means of bindings, to adhere to a strategy. Consequently, we need to differentiate the sets of free variables and free agents of an SL formula $\varphi$. The first contains the variables that are not in a scope of a quantification. The second, instead, contains the agents for which there is no related binding in the scope of a temporal operator. A formula without any free variable (resp., agent) is named *variable-closed* (resp., *agent-closed*). A formula that is both variable- and agent-closed, is named *sentence*. For a given SL formula $\varphi$, by $\mathsf{free}(\varphi)$ we denote the set of both free variables and agents occurring in $\varphi$. The formal definition of $\mathsf{free}(\cdot)$ has been given in [MMPV11, MMPV14]. Just as an example, let $\varphi = \langle\!\langle \mathtt{x} \rangle\!\rangle (\alpha, \mathtt{x})(\beta, \mathtt{y})(\mathtt{Fp})$ be a formula on the agents $\mathrm{Ag} = \{\alpha, \beta, \gamma\}$. Then, we have $\mathsf{free}(\varphi) = \{\gamma, \mathtt{y}\}$, since $\gamma$ is an agent without any binding after $\mathtt{Fp}$ and $\mathtt{y}$ has no quantification at all. Consider also the formulas $(\alpha, \mathtt{z})\varphi$ and $(\gamma, \mathtt{z})\varphi$, where the subformula $\varphi$ is the same as above. Then, we have $\mathsf{free}((\alpha, \mathtt{z})\varphi) = \mathsf{free}(\varphi)$ and $\mathsf{free}((\gamma, \mathtt{z})\varphi) = \{\mathtt{y}, \mathtt{z}\}$, since $\alpha$ is not free in $\varphi$ but $\gamma$ is, *i.e.*, $\alpha \notin \mathsf{free}(\varphi)$ and $\gamma \in \mathsf{free}(\varphi)$. So, $(\gamma, \mathtt{z})\varphi$ is agent-closed while $(\alpha, \mathtt{z})\varphi$ is not.

We now introduce other two general restrictions in which the numbers $|\mathrm{Ag}|$ of agents and $|\mathrm{Vr}|$ of variables that are used to write a formula are fixed to the a priori values $n, m \in [1, \omega[$, respectively. Moreover, we can also forbid the sharing of variables, i.e., each variable is binded to one agent only, so, we cannot force two agents to use the same strategy. We name these three fragments SL[$n$-AG], SL[$m$-VAR], and SL[FVS], respectively.

To make practice with the syntax of SL, we now describe few examples of some game-theoretic properties, which cannot be expressed neither in ATL$^\star$ nor in CHP-SL. We clarify this point later in the paper.

The first we introduce is the well-known concept of *Nash Equilibrium* in concurrent infinite games with Boolean payoffs.

**Example 1.1.6 (Nash Equilibrium)** *Consider the $n$ agents $\alpha_1, \ldots, \alpha_n$ of a game, each of them having, respectively, a possibly different temporal goal described by one of the* LTL *formulas $\psi_1, \ldots, \psi_n$. Then, we can express the existence of a strategy profile $(\mathtt{x}_1, \ldots, \mathtt{x}_n)$ that is a* Nash equilibrium *(NE, for short) for $\alpha_1, \ldots, \alpha_n$* w.r.t. *$\psi_1, \ldots, \psi_n$ by using the* SL *sentence $\varphi_{NE} \triangleq \langle\!\langle \mathtt{x}_1 \rangle\!\rangle \cdots \langle\!\langle \mathtt{x}_n \rangle\!\rangle (\alpha_1, \mathtt{x}_1) \cdots (\alpha_n, \mathtt{x}_n)\ \psi_{NE}$, where $\psi_{NE} \triangleq \bigwedge_{i=1}^{n} (\langle\!\langle \mathtt{y} \rangle\!\rangle (\alpha_i, \mathtt{y})\psi_i) \rightarrow \psi_i$ is a variable-closed formula. Informally, this asserts that every agent $\alpha_i$ has $\mathtt{x}_i$ as one of the best strategy* w.r.t. *the goal $\psi_i$, once all the other strategies of the remaining agents $\alpha_j$, with $j \neq i$, have been fixed to $\mathtt{x}_j$. Note that here we are only considering equilibria under deterministic strategies.*

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

In game theory, it is important to recall that an equilibrium is not always stable. Indeed, there are games like the PD of Example 1.1.2 on page 2 having Nash equilibria that are instable. One of the simplest concepts of stability that is possible to think is called *stability profile*.

**Example 1.1.7 (Stability Profile)** *Think about the same situation of the above example on NE. Then, a* stability profile *(SP, for short) is a strategy profile* $(\mathtt{x}_1, \ldots, \mathtt{x}_n)$ *for* $\alpha_1, \ldots, \alpha_n$ *w.r.t.* $\psi_1, \ldots, \psi_n$ *such that there is no agent* $\alpha_i$ *that can choose a different strategy from* $\mathtt{x}_i$ *without changing its own payoff and penalizing the payoff of another agent* $\alpha_j$*, with* $j \neq i$*. To represent the existence of such a profile, we can use the* SL *sentence* $\varphi_{SP} \triangleq \langle\!\langle \mathtt{x}_1 \rangle\!\rangle (\alpha_1, \mathtt{x}_1) \cdots \langle\!\langle \mathtt{x}_n \rangle\!\rangle (\alpha_n, \mathtt{x}_n)\, \psi_{SP}$*, where* $\psi_{SP} \triangleq \bigwedge_{i,j=1, i \neq j}^{n} \psi_j \rightarrow [\![\mathtt{y}]\!]((\psi_i \leftrightarrow (\alpha_i, \mathtt{y})\psi_i) \rightarrow (\alpha_i, \mathtt{y})\psi_j)$*. Informally, with the* $\psi_{SP}$ *subformula, we assert that, if* $\alpha_j$ *is able to achieve his goal* $\psi_j$*, all strategies* $\mathtt{y}$ *of* $\alpha_i$ *that left unchanged the payoff related to* $\psi_i$*, also let* $\alpha_j$ *to maintain his achieved goal. At this point, it is very easy to ensure the existence of an NE that is also an SP, by using the* SL *sentence* $\varphi_{SNE} \triangleq \langle\!\langle \mathtt{x}_1 \rangle\!\rangle (\alpha_1, \mathtt{x}_1) \cdots \langle\!\langle \mathtt{x}_n \rangle\!\rangle (\alpha_n, \mathtt{x}_n)\, \psi_{SP} \wedge \psi_{NE}$*.*

In a game in which not all agents are peers, we can have one or more of them that may vary the payoff of the others, without having a personal aim, *i.e.*, without looking for the maximization of their own payoffs. Such situations can usually arise when we have games with arbiters or similar characters, like supervisors or government authorities, that have to be fair, *i.e.*, they have to lay down an *equity governance*.

**Example 1.1.8 (Equity Governance)** *Let us think to a game similar to the one described in the previous example, in which there is also a supervisor agent* $\beta$*, which does not have a specific goal. By the way, the peers want him to be fair* w.r.t. *their own goals,* i.e.*, the supervisor has to use a strategy that does not have to prefer one agent over another. This concept is called* equity governance *(EG, for short). In order to formalize it, we can use the* SL *sentence* $\varphi_{EG} \triangleq [\![\mathtt{x}_1]\!] \cdots [\![\mathtt{x}_n]\!] (\alpha_1, \mathtt{x}_1) \cdots (\alpha_n, \mathtt{x}_n) \langle\!\langle \mathtt{y} \rangle\!\rangle (\beta, \mathtt{y})\, \psi_{EG}$*, where* $\psi_{EG} \triangleq \bigwedge_{i,j=1, i<j}^{n} (\langle\!\langle \mathtt{z}_1 \rangle\!\rangle (\beta, \mathtt{z}_1)\psi_i) \wedge (\langle\!\langle \mathtt{z}_2 \rangle\!\rangle (\beta, \mathtt{z}_2)\psi_j) \rightarrow (\psi_i \leftrightarrow \psi_j)$*. Informally, the* $\psi_{EG}$ *subformula asserts that, if there are two strategies* $\mathtt{z}_1$ *and* $\mathtt{z}_2$ *for* $\beta$ *that allow* $\alpha_i$ *and* $\alpha_j$ *to achieve their own goals* $\psi_i$ *and* $\psi_j$*, separately, then the unique strategy* $\mathtt{y}$*previously chosen by the supervisor has to ensures the achievement of either both the goals or none of them. Note that the sentence* $\varphi_{EG}$ *ensure the existence of an EG strategy* $\mathtt{y}$ *for* $\beta$*, in dependence of the strategies* $\mathtt{x}_1, \ldots, \mathtt{x}_n$ *chosen by the peers. To verify the existence of a uniform EG, we may use the* SL *sentence* $\varphi_{UEG} \triangleq \langle\!\langle \mathtt{y} \rangle\!\rangle (\beta, \mathtt{y})\, \psi'_{EG}$*, with* $\psi'_{EG} \triangleq [\![\mathtt{x}_1]\!] \cdot [\![\mathtt{x}_n]\!] (\alpha_1, \mathtt{x}_1) \cdots (\alpha_n, \mathtt{x}_n)\, \psi_{EG}$*, whose difference* w.r.t. $\varphi_{EG}$ *resides only in the alternation of quantifiers. Finally, to verify the existence of a uniform EG that allows also the existence of an NE for the peers, we can use the* SL *sentence* $\varphi_{UEG+NE} \triangleq \langle\!\langle \mathtt{y} \rangle\!\rangle (\beta, \mathtt{y})(\psi'_{EG} \wedge \varphi_{NE})$*.*

Usually, the equity of a supervisor does not ensure that the whole game can advance, *i.e.*, that the peers can achieve their respective goals. Indeed, there are games like the zero-sum ones in which the agents have opposite goals that cannot be achieved at the same time. However, there are different kind of games, as the PPD or the PS of Examples 1.1.3 on page 3 and 1.1.4 on page 4, in which a supervisor can try to help all peers in their intent, by applying an *advancement governance*.

**Example 1.1.9 (Advancement Governance)** *Consider the game described in the previous example of EG. Here, we want to consider an* advancement governance *(AG, for short) for the supervisor,* i.e.*, a strategy for $\beta$ that allows the peers to achieve their own goals, if they have the will and possibility to do so. Formally, this concept can be expressed by using the* SL *sentence $\varphi_{AG} \triangleq [\![\mathtt{x}_1]\!] \cdot [\![\mathtt{x}_n]\!](\alpha_1, \mathtt{x}_1) \cdots (\alpha_n, \mathtt{x}_n)\langle\!\langle \mathtt{y} \rangle\!\rangle(\beta, \mathtt{y})\ \psi_{AG}$, where $\psi_{AG} \triangleq (\bigwedge_{i=1}^{n}(\langle\!\langle \mathtt{z} \rangle\!\rangle(\beta, \mathtt{z})\psi_i) \rightarrow \psi_i)$. Intuitively, the $\psi_{AG}$ subformula expresses the fact that, if $\beta$ has a strategy $\mathtt{z}$ able to force a goal $\psi_i$, once the strategies of the peers $\alpha_1, \ldots, \alpha_n$ have been fixed, then his a priori choice $\mathtt{y}$* w.r.t. *the goals has to force $\psi_i$ as well. As in the case of EG, we can have an uniform version of AG, by using the* SL *sentence $\varphi_{UAG} \triangleq \langle\!\langle \mathtt{y} \rangle\!\rangle(\beta, \mathtt{y})\ \psi'_{AG}$, where $\psi'_{AG} \triangleq [\![\mathtt{x}_1]\!] \cdots [\![\mathtt{x}_n]\!](\alpha_1, \mathtt{x}_1) \cdots (\alpha_n, \mathtt{x}_n)\ \psi_{AG}$.*

Differently from the previous examples, one can consider the case in which the authority agent has his own goal to be satisfied, given the other agents to be in a certain equilibrium. In the context of system design [PR89], *rational synthesis* [FKL10] is a recent improvement of the classical reactive one. In this setting, the adversarial environment is not a monolithic block, but a set of agent components, each of them having their own goal. In the next example, we show that the most typical instances of a rational synthesis problem can be represented in SL.

**Example 1.1.10 (Rational Synthesis)** *Consider a solution concept that is representable in* SL *by means of a suitable formula $\psi_{SC}$,* e.g.*, NE, and a temporal goal $\psi_\beta$ for the system agent. Here, we look for a* rational synthesis *solution for the players,* i.e.*, a strategy profile $(y, x_1, \ldots, x_n)$ such that, if $\beta$ acts according to $y$, then $\psi_\beta$ is satisfied and $(x_1, \ldots, x_n)$ is in an equilibrium according to the solution concept considered,* i.e.*, $\psi_{SC}$ is satisfied. Formally, this concept can be expressed by using the* SL *sentence $\varphi_{RS} = \langle\!\langle y \rangle\!\rangle\langle\!\langle x_1 \rangle\!\rangle \ldots \langle\!\langle x_n \rangle\!\rangle(\beta, y)(a_1, x_1) \ldots (a_n, x_n)(\psi_0 \wedge \psi_{SC})$. As an example, $\psi_{SC}$ can be the formula $\psi_{NE}$ of Example 1.1.6 on page 6. In this case, we obtain the rational synthesis problem for NE.*

### 1.1.3 Basic notions

Before continuing with the formal description of SL, we need to introduce some basic notions related to CGSs, such as those of *track*, *path*, *strategy*, and the like. They are the natural analogous of the ones for Kripke structures and, more generally, for game arenas. All these notions have been already introduced in [MMV10b]. However, for the sake of completeness, as well as for their importance in the definition of SL semantics, we fully report them in this section.

We start with the notions of *track* and *path*. Intuitively, tracks and paths of a CGS are legal sequences of reachable states that can be respectively seen as partial and complete

descriptions of possible outcomes of the game modeled by the structure itself. Formally, 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 $\delta \in \mathrm{Dc}$ such that $(\rho)_{i+1} = \mathsf{tr}((\rho)_i, \delta)$ (resp., $(\pi)_{i+1} = \mathsf{tr}((\pi)_i, \delta)$) [5]. A track $\rho$ is *non-trivial* if it has non-zero length, *i.e.*, $|\rho| > 0$ that is $\rho \neq \epsilon$ [6]. The set $\mathrm{Trk} \subseteq \mathrm{St}^+$ (resp., $\mathrm{Pth} \subseteq \mathrm{St}^\omega$) contains all non-trivial tracks (resp., paths). Moreover, $\mathrm{Trk}(s) \triangleq \{\rho \in \mathrm{Trk} : \mathsf{fst}(\rho) = s\}$ (resp., $\mathrm{Pth}(s) \triangleq \{\pi \in \mathrm{Pth} : \mathsf{fst}(\pi) = s\}$) indicates the subsets of tracks (resp., paths) starting at a state $s \in \mathrm{St}$ [7].

As an example, consider the CGS $\mathcal{G}_{PS}$ in Figure 1.4 on page 4. Then $\rho = s_i \cdot s_1 \cdot {s_1}' \cdot {s_1}' \cdot s_i$ and $\pi = (s_i \cdot s_{1,2} \cdot {s_1}' \cdot s_i \cdot s_{1,2} \cdot {s_2}')^\omega$ are a track and a path, respectively. Moreover, we have that $\mathrm{Trk}_{\mathcal{G}_{PS}} = {\mathrm{St}_{\mathcal{G}_{PS}}}^*$ and $\mathrm{Pth}_{\mathcal{G}_{PS}} = {\mathrm{St}_{\mathcal{G}_{PS}}}^\omega$.

At this point, we can define the concept of *strategy*. Intuitively, a strategy is a scheme for an agent that contains all choices of actions as in dependence of the history of the current outcome. However, observe that here there is not an a priori connection between a strategy and an agent, since the same strategy can be used by more than one agent at the same time. Formally, a *strategy* in a CGS $\mathcal{G}$ 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 $\mathsf{f}$ 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., $\mathrm{Str}(s) \triangleq \mathrm{Trk}(s) \to \mathrm{Ac}$) contains all (resp., $s$-total) strategies.

An example of strategy in the CGS $\mathcal{G}_{PS}$ is given by the function $\mathsf{f}_1 \in \mathrm{Str}$ assigning the action $0$ to all the tracks in which the state $s_i$ occurs an even number of times and the action $1$, otherwise. Another example of strategy is the function $\mathsf{f}_2 \in \mathrm{Str}$ assigning the action $1$ on all possible tracks of the CGS.

We now introduce the notion of *assignment*. Intuitively, an assignment gives a valuation of variables with strategies, where the latter are used to determine the behavior of agents in the game. With more detail, as in the case of first order logic, we use this concept as a technical tool to quantify over strategies associated with variables, independently of agents to which they are related to. So, assignments are used precisely as a way to define a correspondence between variables and agents via strategies.

**Definition 1.1.3 (Assignments)** *An* assignment *in a* CGS $\mathcal{G}$ *is a partial function* $\chi : \mathrm{Vr} \cup \mathrm{Ag} \rightharpoonup \mathrm{Str}$ *mapping variables and agents in its domain to a strategy. An assignment* $\chi$ *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* $\chi$ *is* $s$-total *if all strategies* $\chi(l)$ *are* $s$*-total, for* $l \in \mathsf{dom}(\chi)$*. The set* $\mathrm{Asg} \triangleq \mathrm{Vr} \cup \mathrm{Ag} \rightharpoonup \mathrm{Str}$ *(resp.,* $\mathrm{Asg}(s) \triangleq \mathrm{Vr} \cup \mathrm{Ag} \rightharpoonup \mathrm{Str}(s)$*) contains all (resp., $s$-total) assignments. Moreover,* $\mathrm{Asg}(\mathrm{X}) \triangleq \mathrm{X} \to \mathrm{Str}$ *(resp.,* $\mathrm{Asg}(\mathrm{X}, s) \triangleq \mathrm{X} \to \mathrm{Str}(s)$*) indicates the subset of* X-defined *(resp., $s$-total) assignments,* i.e.*, (resp., $s$-total) assignments defined on the set* $\mathrm{X} \subseteq \mathrm{Vr} \cup \mathrm{Ag}$*.*

As an example of assignment, consider the function $\chi_1 \in \mathrm{Asg}$ in the CGS $\mathcal{G}_{PS}$, with $\mathsf{dom}(\chi_1) = \{\mathsf{P}_1, \mathsf{x}\}$, such that $\chi_1(\mathsf{P}_1) = \mathsf{f}_1$ and $\chi_1(\mathsf{x}) = \mathsf{f}_2$. As another example, consider the assignment $\chi_2 \in \mathrm{Asg}$ in the same CGS, with $\mathsf{dom}(\chi_2) = \mathrm{Ag}$, such that $\chi_2(\mathsf{S}) = \mathsf{f}_1$ and $\chi_2(\mathsf{P}_1) = \chi_2(\mathsf{P}_2) = \mathsf{f}_2$. Note that $\chi_2$ is complete, while $\chi_1$ is not.

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

[6] The Greek letter $\epsilon$ stands for the *empty sequence*.

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

Given an assignment $\chi \in \mathrm{Asg}$, an agent or variable $l \in \mathrm{Vr} \cap \mathrm{Ag}$, and a strategy $\mathsf{f} \in \mathrm{Str}$, we need to describe the *redefinition* of $\chi$, *i.e.*, a new assignment equal to the first one on all elements of its domain but $l$, on which it assumes the value $\mathsf{f}$. Formally, with $\chi[l \mapsto \mathsf{f}] \in \mathrm{Asg}$ we denote the new assignment defined on $\mathsf{dom}(\chi[l \mapsto \mathsf{f}]) \triangleq \mathsf{dom}(\chi) \cup \{l\}$ that returns $\mathsf{f}$ on $l$ and is equal to $\chi$ on the remaining part of its domain, *i.e.*, $\chi[l \mapsto \mathsf{f}](l) \triangleq \mathsf{f}$ and $\chi[l \mapsto \mathsf{f}](l') \triangleq \chi(l')$, for all $l' \in \mathsf{dom}(\chi) \setminus \{l\}$. Intuitively, if we have to add or update a strategy that needs to be bound by an agent or variable, we can simply take the old assignment and redefine it by using the above notation. It is worth observing that, if $\chi$ and $\mathsf{f}$ are $s$-total then $\chi[l \mapsto \mathsf{f}]$ is $s$-total, as well.

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

**Definition 1.1.4 (Plays)** *A path $\pi \in \mathrm{Pth}(s)$ starting at a state $s \in \mathrm{St}$ is a* play w.r.t. *a complete $s$-total assignment $\chi \in \mathrm{Asg}(s)$ ($(\chi, s)$-*play*, for short) if, for all $i \in \mathbb{N}$, it holds that $(\pi)_{i+1} = \mathsf{tr}((\pi)_i, \delta)$, where $\delta(a) \triangleq \chi(a)((\pi)_{\leq i})$, for each $a \in \mathrm{Ag}$* [8]*. The partial function $\mathsf{play} : \mathrm{Asg} \times \mathrm{St} \rightharpoonup \mathrm{Pth}$, with $\mathsf{dom}(\mathsf{play}) \triangleq \{(\chi, s) : \mathrm{Ag} \subseteq \mathsf{dom}(\chi) \wedge \chi \in \mathrm{Asg}(s) \wedge s \in \mathrm{St}\}$, returns the $(\chi, s)$-play $\mathsf{play}(\chi, s) \in \mathrm{Pth}(s)$, for all pairs $(\chi, s)$ in its domain.*

As last example, consider again the CGS $\mathcal{G}_{PS}$ and the complete $s_0$-total assignment $\chi_2$ defined above. Then, we have that $\mathsf{play}(\chi_2, s_0) = (s_i \cdot s_{1,2} \cdot {s_1}' \cdot s_i \cdot s_{1,2} \cdot {s_2}')^\omega$.

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

**Definition 1.1.5 (Global Translation)** *For a given state $s \in \mathrm{St}$ and a complete $s$-total assignment $\chi \in \mathrm{Asg}(s)$, the $i$-*th global translation *of $(\chi, s)$, with $i \in \mathbb{N}$, is the pair of a complete assignment and a state $(\chi, s)^i \triangleq ((\chi)_{(\pi)_{\leq i}}, (\pi)_i)$, where $\pi = \mathsf{play}(\chi, s)$ and $(\chi)_{(\pi)_{\leq i}}$ denotes the $(\pi)_i$-total assignment, with $\mathsf{dom}((\chi)_{(\pi)_{\leq i}}) = \mathsf{dom}(\chi)$, such that, for all $l \in \mathsf{dom}(\chi)$, $\mathsf{dom}((\chi)_{(\pi)_{\leq i}}(l)) = \{\rho \in \mathrm{Trk}((\pi)_i) : (\pi)_{\leq i} \cdot \rho \in \mathsf{dom}(\chi(l))\}$ and $(\chi)_{(\pi)_{\leq i}}(l)(\rho) = \chi(l)((\pi)_{\leq i} \cdot \rho)$, for all $\rho \in \mathsf{dom}((\chi)_{(\pi)_{\leq i}}(l))$.*

### 1.1.4 Semantics

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

**Definition 1.1.6 (SL Semantics)** *Given a* CGS $\mathcal{G}$*, for all* SL *formulas $\varphi$, 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.*

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

1. $\mathcal{G}, \chi, s \models p$ *if* $p \in \mathsf{ap}(s)$, *with* $p \in \mathrm{AP}$.

2. *For all formulas* $\varphi$, $\varphi_1$, *and* $\varphi_2$, *it holds that:*

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

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

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

3. *For a variable* $x \in \mathrm{Vr}$ *and a formula* $\varphi$, *it holds that:*

   (a) $\mathcal{G}, \chi, s \models \langle\!\langle x \rangle\!\rangle \varphi$ *if there is 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* $x \in \mathrm{Vr}$, *and a formula* $\varphi$, *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* $\chi$ *is complete, for all formulas* $\varphi$, $\varphi_1$, *and* $\varphi_2$, *it holds that:*

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

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

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

In order to complete the description of the semantics, we now give the classic notions of *model* and *satisfiability* of an SL sentence. We say that a CGS $\mathcal{G}$ is a *model* of an SL sentence $\varphi$, in symbols $\mathcal{G} \models \varphi$, if $\mathcal{G}, \varnothing, s_{\mathrm{o}} \models \varphi$. [9] In general, we also say that $\mathcal{G}$ is a *model* for $\varphi$ on $s \in \mathrm{St}$, in symbols $\mathcal{G}, s \models \varphi$, if $\mathcal{G}, \varnothing, s \models \varphi$. An SL sentence $\varphi$ is *satisfiable* if there is a model for it.

It remains to formalize the concepts of *implication* and *equivalence* between SL formulas, which are useful to describe transformations preserving the meaning of a specification. Given two SL formulas $\varphi_1$ and $\varphi_2$, with $\mathsf{free}(\varphi_1) = \mathsf{free}(\varphi_2)$, we say that $\varphi_1$ *implies* $\varphi_2$, 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 $\varphi_1$ is *equivalent* to $\varphi_2$, in symbols $\varphi_1 \equiv \varphi_2$, if both $\varphi_1 \Rightarrow \varphi_2$ and $\varphi_2 \Rightarrow \varphi_1$ hold.

In the rest of the paper, especially when we describe a decision procedure, we may consider formulas in *existential normal form* (*enf*, for short) and *positive normal form* (*pnf*, for short),

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

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

At this point, in order to better understand the meaning of the SL semantics, we discuss some examples of formulas interpreted over the CGSs previously described.

We start by explaining how a strategy can be shared by different agents.

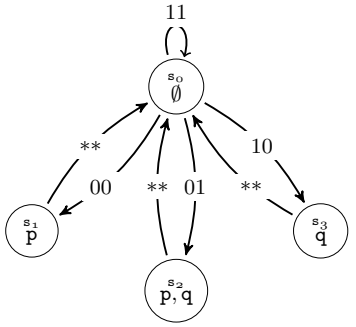


Figure 1.6: The CGS $\mathcal{G}_{SV}$.

**Example 1.1.11 (Shared Variable)** *Consider the* SL *sentence* $\varphi = \langle\!\langle \mathtt{x} \rangle\!\rangle [\![ \mathtt{y} ]\!] \langle\!\langle \mathtt{z} \rangle\!\rangle ((\alpha, \mathtt{x})(\beta, \mathtt{y})(\mathtt{Xp}) \wedge (\alpha, \mathtt{y})(\beta, \mathtt{z})(\mathtt{Xq}))$. *It is immediate to note that both agents* $\alpha$ *and* $\beta$ *use the strategy associated with* $\mathtt{y}$ *to achieve simultaneously the* LTL *temporal goals* $\mathtt{Xp}$ *and* $\mathtt{Xq}$. *A model for* $\varphi$ *is given by the* CGS $\mathcal{G}_{SV} \triangleq \langle \{\mathtt{p}, \mathtt{q}\}, \{\alpha, \beta\}, \{0,1\}, \{\mathtt{s}_0, \mathtt{s}_1, \mathtt{s}_2, \mathtt{s}_3\}, \mathtt{L}, \mathtt{tr}, \mathtt{s}_0 \rangle$, *where* $\mathtt{L}(\mathtt{s}_0) \triangleq \emptyset$, $\mathtt{L}(\mathtt{s}_1) \triangleq \{\mathtt{p}\}$, $\mathtt{L}(\mathtt{s}_2) \triangleq \{\mathtt{p}, \mathtt{q}\}$, $\mathtt{L}(\mathtt{s}_3) \triangleq \{\mathtt{q}\}$, $\mathtt{tr}(\mathtt{s}_0, (0,0)) \triangleq \mathtt{s}_1$, $\mathtt{tr}(\mathtt{s}_0, (0,1)) \triangleq \mathtt{s}_2$, $\mathtt{tr}(\mathtt{s}_0, (1,0)) \triangleq \mathtt{s}_3$, *and all the remaining transitions (with any decision) go to* $\mathtt{s}_0$. *In Figure 1.6, we report a graphical representation of the structure. Clearly,* $\mathcal{G}_{SV} \models \varphi$ *by letting, on* $\mathtt{s}_0$, *the variables* $\mathtt{x}$ *to chose action* 0 *(the goal* $(\alpha, \mathtt{x})(\beta, \mathtt{y})(\mathtt{Xp})$ *is satisfied for any choice of* $\mathtt{y}$, *since we can move from* $\mathtt{s}_0$ *to either* $\mathtt{s}_1$ *or* $\mathtt{s}_2$, *both labeled with* $\mathtt{p}$*) and* $\mathtt{z}$ *to choose action* 1 *when* $\mathtt{y}$ *has action* 0 *and, vice versa,* 0 *when* $\mathtt{y}$ *has* 1 *(in both cases, the goal* $(\alpha, \mathtt{y})(\beta, \mathtt{z})(\mathtt{Xq})$ *is satisfied, since one can move from* $\mathtt{s}_0$ *to either* $\mathtt{s}_2$ *or* $\mathtt{s}_3$, *both labeled with* $\mathtt{q}$*).*

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

**Example 1.1.12 (Equilibrium Profiles)** *Let us first consider the* CGS $\mathcal{G}_{PD}$ *of the prisoner's dilemma described in the Example 1.1.2 on page 2. Intuitively, each of the two accomplices* $\mathtt{A}_1$ *and* $\mathtt{A}_2$ *want to avoid the prison. These goals can be represented by the* LTL *formulas* $\psi_{\mathtt{A}_1} \triangleq \mathtt{Gf}_{\mathtt{A}_1}$ *and* $\psi_{\mathtt{A}_2} \triangleq \mathtt{Gf}_{\mathtt{A}_2}$, *respectively. The existence of a Nash equilibrium in* $\mathcal{G}_{PD}$ *for the two accomplices w.r.t. the above goals can be written as* $\phi_{NE} \triangleq \langle\!\langle \mathtt{x}_1 \rangle\!\rangle (\mathtt{A}_1, \mathtt{x}_1) \langle\!\langle \mathtt{x}_2 \rangle\!\rangle (\mathtt{A}_2, \mathtt{x}_2)\, \psi_{NE}$, *where* $\psi_{NE} \triangleq ((\langle\!\langle \mathtt{y} \rangle\!\rangle (\mathtt{A}_1, \mathtt{y}) \psi_{\mathtt{A}_1}) \rightarrow \psi_{\mathtt{A}_1}) \wedge ((\langle\!\langle \mathtt{y} \rangle\!\rangle (\mathtt{A}_2, \mathtt{y}) \psi_{\mathtt{A}_2}) \rightarrow \psi_{\mathtt{A}_2})$, *which results to be an instantiation of the general sentence* $\varphi_{NE}$ *of Example 1.1.6 on page 6. In the same way, the existence of a stable Nash equilibrium can be represented with the sentence* $\phi_{SNE} \triangleq \langle\!\langle \mathtt{x}_1 \rangle\!\rangle (\mathtt{A}_1, \mathtt{x}_1) \langle\!\langle \mathtt{x}_2 \rangle\!\rangle (\mathtt{A}_2, \mathtt{x}_2)\, \psi_{NE} \wedge \psi_{SP}$, *where* $\psi_{SP} \triangleq (\psi_1 \rightarrow [\![ \mathtt{y} ]\!]((\psi_2 \leftrightarrow (\mathtt{A}_2, \mathtt{y})\psi_2) \rightarrow (\mathtt{A}_2, \mathtt{y})\psi_1)) \wedge (\psi_2 \rightarrow [\![ \mathtt{y} ]\!]((\psi_1 \leftrightarrow (\mathtt{A}_1, \mathtt{y})\psi_1) \rightarrow (\mathtt{A}_1, \mathtt{y})\psi_2))$, *which is a particular case of the sentence* $\varphi_{SNE}$ *of Example 1.1.7 on page 7. Now, it is easy to see that* $\mathcal{G}_{PD} \models \phi_{SNE}$ *and, so,* $\mathcal{G}_{PD} \models \phi_{NE}$. *Indeed, an assignment* $\chi \in \mathrm{Asg}_{\mathcal{G}_{PD}}(\mathrm{Ag}, \mathtt{s}_i)$, *for which* $\chi(\mathtt{A}_1)(\mathtt{s}_i) = \chi(\mathtt{A}_2)(\mathtt{s}_i) = \mathtt{D}$, *is a stable equilibrium profile, i.e., it is such that* $\mathcal{G}_{PD}, \chi, \mathtt{s}_i \models \psi_{NE} \wedge \psi_{SP}$. *This is due to the fact that, if an agent* $\mathtt{A}_k$, *for* $k \in \{1,2\}$, *choses another strategy* $\mathtt{f} \in \mathrm{Str}_{\mathcal{G}_{PD}}(\mathtt{s}_i)$, *he is still unable to achieve his goal* $\psi_k$, *i.e.,* $\mathcal{G}_{PD}, \chi^{\mathtt{A}_k \mapsto \mathtt{f}}, \mathtt{s}_i \not\models \psi_k$, *so, he*

*cannot improve his payoff. Moreover, this equilibrium is stable, since the payoff of an agent cannot be made worse by the changing of the strategy of the other agent. However, it is interesting to note that there are instable equilibria too. One of these is represented by the assignment* $\chi' \in \mathrm{Asg}_{\mathcal{G}_{PD}}(\mathrm{Ag}, \mathtt{s}_i)$*, for which* $\chi'(\mathtt{A}_1)(\mathtt{s}_i{}^j) = \chi'(\mathtt{A}_2)(\mathtt{s}_i{}^j) = \mathtt{C}$*, for all* $j \in \mathbb{N}$*. Indeed, we have that* $\mathcal{G}_{PD}, \chi', \mathtt{s}_i \models \psi_{NE}$*, since* $\mathcal{G}_{PD}, \chi', \mathtt{s}_i \models \psi_1$ *and* $\mathcal{G}_{PD}, \chi', \mathtt{s}_i \models \psi_2$*, but* $\mathcal{G}_{PD}, \chi', \mathtt{s}_i \not\models \psi_{SP}$*. The latter property holds because, if one of the agents* $\mathtt{A}_k$*, for* $k \in \{1, 2\}$*, choses a different strategy* $\mathtt{f}' \in \mathrm{Str}_{\mathcal{G}_{PD}}(\mathtt{s}_i)$ *for which there is a* $j \in \mathbb{N}$ *such that* $\mathtt{f}'(\mathtt{s}_i{}^j) = \mathtt{D}$*, he cannot improve his payoff but makes surely worse the payoff of the other agent, i.e.,* $\mathcal{G}_{PD}, \chi'[\mathtt{A}_k \mapsto \mathtt{f}'], \mathtt{s}_i \models \psi_k$ *but* $\mathcal{G}_{PD}, \chi'[\mathtt{A}_k \mapsto \mathtt{f}'], \mathtt{s}_i \not\models \psi_{3-k}$*. Finally, consider the* CGS $\mathcal{G}_{PRS}$ *of the paper, rock, and scissor game described in the Example 1.1.1 on page 2 together with the associated formula for the Nash equilibrium* $\phi_{NE} \triangleq \langle\!\langle \mathtt{x}_1 \rangle\!\rangle(\mathtt{A}, \mathtt{x}_1)\langle\!\langle \mathtt{x}_2 \rangle\!\rangle(\mathtt{B}, \mathtt{x}_2)\, \psi_{NE}$*, where* $\psi_{NE} \triangleq ((\langle\!\langle \mathtt{y} \rangle\!\rangle(\mathtt{A}, \mathtt{y})\psi_{\mathtt{A}}) \rightarrow \psi_{\mathtt{A}}) \wedge ((\langle\!\langle \mathtt{y} \rangle\!\rangle(\mathtt{B}, \mathtt{y})\psi_{\mathtt{B}}) \rightarrow \psi_{\mathtt{B}})$ *with* $\psi_{\mathtt{A}} \triangleq \mathtt{Fw}_{\mathtt{A}}$ *and* $\psi_{\mathtt{B}} \triangleq \mathtt{Fw}_{\mathtt{B}}$ *representing the* LTL *temporal goals for Alice and Bob, respectively. Then, it is not hard to see that* $\mathcal{G}_{PRS} \not\models \phi_{NE}$*, i.e., there are no Nash equilibria in this game, since there is necessarily an agent that can improve his/her payoff by changing his/her strategy.*

**Example 1.1.13 (Law and Order)** *Consider the* CGS $\mathcal{G}_{PPD}$ *given in Example 1.1.3 on page 3. It is easy to see that Police can ensure at least one prisoner to be definitely in jail. Indeed the formula* $\varphi_1 = \langle\!\langle y \rangle\!\rangle [\![ x_1 ]\!] [\![ x_2 ]\!] (\mathtt{P}, y)(\mathtt{A}_1, x_1)(\mathtt{A}_2, x_2)((\mathtt{FG}\neg \mathtt{f}_{\mathtt{A}_1}) \vee (\mathtt{FG}\neg \mathtt{f}_{\mathtt{A}_2}))$ *is satisfied over* $\mathcal{G}_{PPD}$*. A way to see this is to consider the strategy* $\mathtt{f}$ *for* $\mathtt{P}$ *given by* $\mathtt{f}(\rho) = 0$*, for all* $\rho \in \mathrm{Trk}$*, which allows agent* $\mathtt{P}$ *to always avoid the state* $s_{\mathtt{A}_1,\mathtt{A}_2}$*. On the other hand, the formula* $\varphi_2 = [\![ x_1 ]\!] [\![ x_2 ]\!] \langle\!\langle y \rangle\!\rangle (\mathtt{P}, y)(\mathtt{A}_1, x_1)(\mathtt{A}_2, x_2)(\mathtt{FG}(\neg \mathtt{f}_{\mathtt{A}_1} \wedge \neg \mathtt{f}_{\mathtt{A}_2}))$ *is not satisfied over* $\mathcal{G}_{PPD}$*. Indeed, if agents* $\mathtt{A}_1$ *and* $\mathtt{A}_2$ *use strategies* $\mathtt{f}_1$ *and* $\mathtt{f}_2$*, respectively, such that* $\mathtt{f}_2(s_i) = 1 - \mathtt{f}_1(s_i)$*, we have that, whatever agent* $\mathtt{P}$ *does, at least one of them gains freedom.*

**Example 1.1.14 (Fair Scheduler)** *Consider the* CGS $\mathcal{G}_{PS}$ *of Example 1.1.4 on page 4 and suppose the Scheduler wants to ensure that, whatever process makes a request, the resource is eventually granted to it. We can represent this specification by means of the formula* $\varphi = \langle\!\langle y \rangle\!\rangle [\![ x_1 ]\!] [\![ x_2 ]\!] (\mathtt{S}, y)(\mathtt{P}_1, x_1)(\mathtt{P}_2, x_2)(\mathtt{G}((\mathtt{r}_1 \rightarrow \mathtt{Fg}_1) \wedge (\mathtt{r}_2 \rightarrow \mathtt{Fg}_2)))$*. It is easy to see that* $\mathcal{G}_{PS} \models \varphi$*. Indeed, consider the strategy* $\mathtt{f}$ *for* $\mathtt{S}$ *defined as follows. For all tracks of the form* $\rho \cdot s'_i$*, we set a possible preemption,* i.e.*,* $\mathtt{f}(\rho \cdot s'_i) = 1$*. For the tracks of the form* $\rho \cdot s_{1,2}$ *we set the action as prescribed in the sequel:* (i) *if there is no occurrence of* $s'_1$ *and* $s'_2$ *or the last occurrence is* $s'_1$*, then we release the resource to* $\mathtt{P}_1$*,* i.e.*,* $\mathtt{f}(\rho \cdot s_{1,2}) = 0$*;* (ii) *if the last occurrence in* $\rho$ *between* $s'_1$ *and* $s'_2$ *is* $s'_1$*, then then we release the resource to* $\mathtt{P}_2$*,* i.e.*,* $\mathtt{f}(\rho \cdot s_{1,2}) = 1$.

In the next example, we show the expressive power of alternation of quantifications in SL by pointing out an example based on the "Romeo and Juliet" game, shown in Example 1.1.5 on page 4.

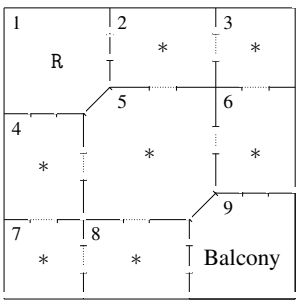



Figure 1.7: Winning position.

**Example 1.1.15 (Quantification modalities.)** *First consider the* CGS $\mathcal{G}_{RJ}$ *described in Example 1.1.5 on page 4. Romeo* $\mathtt{R}$*, helped by Shakespeare* $\mathtt{S}$*, wants to reach the*

*balcony of his loved Juliet while never meeting the Montagues family* $\mathtt{M}$ *nor the Capulets family* $\mathtt{C}$*. According to the* CGS*, this aim can be easily expressed with the* LTL *formula* $\psi = \mathtt{Fr} \wedge \mathtt{G}\neg\mathtt{c}$*. The game can be played with several quantification modalities, each of them having a different meaning. For example, we can consider the formula* $\varphi_1 = [\![\mathtt{z}]\!][\![\mathtt{w}]\!]\langle\!\langle x\rangle\!\rangle\langle\!\langle \mathtt{y}\rangle\!\rangle \flat\psi$*, where* $\flat = (\mathtt{R}, \mathtt{x})(\mathtt{S}, \mathtt{y})(\mathtt{M}, \mathtt{z})(\mathtt{C}, \mathtt{w})$*. This means that both* $\mathtt{R}$ *and* $\mathtt{S}$ *have full visibility about the moves of the families in the house. Conversely, we can consider the formula* $\varphi_2 = \langle\!\langle x\rangle\!\rangle\langle\!\langle \mathtt{y}\rangle\!\rangle[\![\mathtt{z}]\!][\![\mathtt{w}]\!]\flat\psi$*, which means that both* $\mathtt{R}$ *and* $\mathtt{S}$ *have to play a strategy that is winning no matter what the families do. Moreover, one can select a modality in which the alternation of the quantification is greater than one. For instance, in the formula* $\varphi_3 = \langle\!\langle x\rangle\!\rangle[\![\mathtt{z}]\!][\![\mathtt{w}]\!]\langle\!\langle \mathtt{y}\rangle\!\rangle\flat\psi$*, Romeo has to play independently on the families while Shakespeare can select a strategy according to what the families are doing in the house. In other word, a team of agents has visibility on the adversary team that differs agent by agent, and depends on the order in which their strategies are quantified in the formula. It is worth noting that, while* $\varphi_1$ *and* $\varphi_2$ *can be represented also in* $\mathrm{ATL}^\star$*,* $\varphi_3$ *does not have an equivalent* $\mathrm{ATL}^\star$ *formula, due to its alternation number of quantifications equal to* $2$*.*

*It is easy to see that the position in Figure 1.7 on the previous page, with rooms* $1$ *and* $9$ *completely isolated and families standing in the remaining sector of the house composed by rooms from* $2$ *to* $8$*, is winning for Romeo and Shakespeare. In fact, a trivial strategy for them is given by Romeo remaining in the room* $1$ *until Shakespeare closes the doors needed to block the families in two rooms (possibly the same) of the house, then opening a path toward the balcony for Romeo. Note that such a path always exists, since there are three different non intersecting paths from room* $1$ *to room* $9$ *(i.e.,* $1-2-3-6-9$*,* $1-4-7-8-9$*, and* $1-5-9$*) and the families can occupy at most two of them.*

*Moreover, it is easy to see that the winning position in Figure 1.7 on the preceding page cannot be reached if Shakespeare does not have full visibility on families' moves. Indeed, starting from the initial position, Shakespeare is forced to close doors* $(1,2)$ *and* $(1,4)$*, in order to prevent Romeo to be shot at the second step, since Montagues and Capulets can move in* $2$ *and* $4$*, respectively. But, by closing those doors, Montagues (resp., Capulets) can move in* $6$ *(resp.,* $8$*), being able to reach the balcony room in the next step of the game then holding there indefinitely. This makes impossible to eventually satisfy* $\mathtt{r}$ *without satisfying* $\mathtt{c}$*, as well. Hence, we have that* $\mathcal{G}_{RJ} \not\models \varphi_2$*.*

*On the other hand, if Shakespeare knows where the families are moving from the initial state, then he can close the nearest doors and prevent them to reach the key rooms, i.e.,* $1$ *and* $9$*, in the next steps. Indeed, if Montagues and Capulets are going to move in* $2$ *and* $4$*, then Shakespeare closes the doors* $(1,2)$ *and* $(1,4)$*, otherwise, if Montagues and Capulets are going to move in* $6$ *and* $8$*, then Shakespeare closes the doors* $(6,9)$ *and* $(7,9)$*. Again, the "mixed" strategies of Montagues and Capulets given by moving in* $2$ *and* $8$ *or* $4$ *and* $6$ *can be neutralized by a suitable choice for Shakespeare. Moreover, it is easy to see that, after a well played initial step, the the winning position of Figure 1.7 on the previous page is easily reachable by Romeo and Shakespeare, as the latter can close the doors* $(1,5)$ *and* $(5,9)$ *at the second step and then close the remaining doors in order to isolate the rooms* $1$ *and* $9$*.*

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

Finally, we discuss another examples that is useful to point out the power of forcing two or more agents to share a strategy. Hex is a two-player game, red 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 as in the following example.

**Example 1.1.16 (Stealing-Strategy argument in Hex)** *First model Hex with a* CGS $\mathcal{G}_H$ *whose states represent every possible configuration reached during a play between Player "*$\mathtt{r}$*" red and "*$\mathtt{b}$*" blue. Then, verify the negation of the stealing-strategy argument by checking* $\mathcal{G}_H \models \langle\langle \mathtt{x} \rangle\rangle (\mathtt{r}, \mathtt{x})(\mathtt{b}, \mathtt{x})(\mathsf{Fcnc}_{\mathtt{r}})$. *Intuitively, this sentence says that agent* $\mathtt{r}$ *has a strategy that, once it is copied (bound) by* $\mathtt{b}$ *it allows the former to win, i.e., to be the first to connect the related red edges (*$\mathsf{Fcnc}_{\mathtt{r}}$*).*

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

## 1.2 Model-Checking Hardness

In this section, we show the non-elementary lower bound for the model-checking problem of SL. Precisely, we prove that, for sentences having alternation number $k$, this problem is $k$-EXPSPACE-HARD. To this aim, in Subsection 1.2.1, we first recall syntax and semantics of *Quantified propositional temporal logic* [Sis83]. Then, in Subsection 1.2.2, we give a reduction from the satisfiability problem for this logic to the model-checking problem for SL.

### 1.2.1 Quantified propositional temporal logic

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

**Definition 1.2.1 (**QPTL **Syntax)** QPTL formulas *are built inductively from the sets of atomic propositions* AP*, by using the following grammar, where* $p \in \mathrm{AP}$*:*

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

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

## 1.2. Model-Checking Hardness

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

In order to define the semantics of QPTL, we have first to introduce the concepts of truth evaluations used to interpret the meaning of atomic propositions at the passing of time. A *temporal truth evaluation* is a function $\mathtt{tte} : \mathbb{N} \to \{\mathtt{f}, \mathtt{t}\}$ that maps each natural number to a Boolean value. Moreover, a *propositional truth evaluation* is a partial function $\mathtt{pte} : \mathrm{AP} \rightharpoonup \mathrm{TTE}$ mapping every atomic proposition in its domain to a temporal truth evaluation. The sets $\mathrm{TTE} \triangleq \mathbb{N} \to \{\mathtt{f}, \mathtt{t}\}$ and $\mathrm{PTE} \triangleq \mathrm{AP} \rightharpoonup \mathrm{TTE}$ contain, respectively, all temporal and propositional truth evaluations.

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

**Definition 1.2.2 (QPTL Semantics)** *For all* QPTL *formulas $\varphi$, propositional truth evaluation $\mathsf{pte} \in \mathrm{PTE}$ with $\mathsf{free}(\varphi) \subseteq \mathsf{dom}(\mathsf{pte})$, and numbers $k \in \mathbb{N}$, the modeling relation $\mathsf{pte}, k \models \varphi$ is inductively defined as follows.*

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

Obviously, a QPTL sentence $\varphi$ is *satisfiable* if $\varnothing, 0 \models \varphi$. Observe that the described semantics is slightly different but completely equivalent to that proposed and used in [SVW87] to prove the non-elementary hardness result for the satisfiability problem.

### 1.2.2 Non-elementary lower-bound

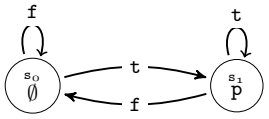



Figure 1.8: The CGS $\mathcal{G}_{Rdc}$.

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

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

**Lemma 1.2.1 (**QPTL **Reduction)** *There is a one-agent* CGS $\mathcal{G}_{Rdc}$ *such that, for each* QPTL$[k$-ALT$]$ *sentence* $\varphi$*, with* $k \in \mathbb{N}$*, there exists an* SL *variable-closed formula* $\overline{\varphi}$ *such that* $\varphi$ *is satisfiable iff* $\mathcal{G}_{Rdc}, \chi, s_{\mathrm{o}} \models \overline{\varphi}$*, for all complete assignments* $\chi \in \mathrm{Asg}(\mathrm{Ag}, s_{\mathrm{o}})$.

*Proof.* Consider the one-agent CGS $\mathcal{G}_{Rdc} \triangleq \langle \{\mathtt{p}\}, \{\alpha\}, \{\mathtt{f}, \mathtt{t}\}, \{\mathtt{s}_{\mathrm{o}}, \mathtt{s}_{1}\}, \mathsf{L}, \mathtt{tr}, \mathtt{s}_{\mathrm{o}} \rangle$ depicted in Figure 1.8, where the two actions are the Boolean values false and true and where the labeling and transition functions $\mathsf{L}$ and $\mathtt{tr}$ are set as follows: $\mathsf{L}(\mathtt{s}_{\mathrm{o}}) \triangleq \emptyset$, $\mathsf{L}(\mathtt{s}_{1}) \triangleq \{\mathtt{p}\}$, and $\mathtt{tr}(s, \mathsf{dc}) = \mathtt{s}_{\mathrm{o}}$ iff $\mathsf{dc}(\alpha) = \mathtt{f}$, for all $s \in \mathrm{St}$ and $\mathsf{dc} \in \mathrm{Dc}$. Moreover, consider the transformation function $\overline{\cdot} : \text{QPTL} \to \text{SL}$ inductively defined as follows:

- $\overline{q} \triangleq (\alpha, \mathtt{x}_q)\mathtt{Xp}$, for $q \in \mathrm{AP}$;
- $\overline{\exists q.\varphi} \triangleq \langle\!\langle \mathtt{x}_q \rangle\!\rangle \overline{\varphi}$;
- $\overline{\forall q.\varphi} \triangleq [\![ \mathtt{x}_q ]\!] \overline{\varphi}$;
- $\overline{\mathtt{O}\mathtt{p}\varphi} \triangleq \mathtt{O}\mathtt{p}\overline{\varphi}$, where $\mathtt{O}\mathtt{p} \in \{\neg, \mathtt{X}, \mathtt{F}, \mathtt{G}\}$;
- $\overline{\varphi_1 \mathtt{O}\mathtt{p} \varphi_2} \triangleq \overline{\varphi_1} \mathtt{O}\mathtt{p} \overline{\varphi_2}$, where $\mathtt{O}\mathtt{p} \in \{\wedge, \vee\}$.

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

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

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

- $\psi = q$.

  By Item 1 of Definition 1.2.2 on the preceding page of QPTL semantics, we have that $\mathsf{pte}, i \models q$ iff $\mathsf{pte}(q)(i) = \mathtt{t}$. Thus, due to the above constraint on the assignment, it follows that $\mathsf{pte}, i \models q$ iff $\chi(\mathtt{x}_q)((\pi)_{\leq i}) = \mathtt{t}$. Now, by applying Items 4 and 5a of Definition 1.1.6 on page 10 of SL semantics, we have that $\mathcal{G}_{Rdc}, (\chi, \mathtt{s}_{\mathrm{o}})^i \models (\alpha, \mathtt{x}_q)\mathtt{Xp}$ iff $\mathcal{G}_{Rdc}, (\chi'[\alpha \mapsto \chi'(\mathtt{x}_q)], s')^1 \models \mathtt{p}$, where $(\chi', s') = (\chi, \mathtt{s}_{\mathrm{o}})^i$. At this point, due to the particular structure of the CGS $\mathcal{G}_{Rdc}$, we have that $\mathcal{G}_{Rdc}, (\chi'[\alpha \mapsto \chi'(\mathtt{x}_q)], s')^1 \models \mathtt{p}$ iff $(\pi')_1 = \mathtt{s}_1$, where $\pi' \triangleq \mathsf{play}(\chi'[\alpha \mapsto \chi'(\mathtt{x}_q)], s')$, which in turn is equivalent to $\chi'(\mathtt{x}_q)((\pi')_{\leq 0}) = \mathtt{t}$. So, $\mathcal{G}_{Rdc}, (\chi, \mathtt{s}_{\mathrm{o}})^i \models (\alpha, \mathtt{x}_q)\mathtt{Xp}$ iff $\chi'(\mathtt{x}_q)((\pi')_{\leq 0}) = \mathtt{t}$. Now,

by observing that $(\pi')_{\leq 0} = (\pi)_i$ and using the above definition of $\chi'$, we obtain that $\chi'(\mathtt{x}_q)((\pi')_{\leq 0}) = \chi(\mathtt{x}_q)((\pi)_{\leq i})$. Hence, $\mathsf{pte}, i \models q$ iff $\mathsf{pte}(q)(i) = \chi(\mathtt{x}_q)((\pi)_{\leq i}) = \mathtt{t} = \chi'(\mathtt{x}_q)((\pi')_{\leq 0})$ iff $\mathcal{G}_{Rdc}, (\chi, \mathtt{s}_0)^i \models (\alpha, \mathtt{x}_q)\mathtt{Xp}$.

- $\psi = \exists q.\psi'$.

  *[Only if].* If $\mathsf{pte}, i \models \exists q.\psi'$, by Item 3a of Definition 1.2.2 on page 16, there exists a temporal truth evaluation $\mathsf{tte} \in \mathrm{TTE}$ such that $\mathsf{pte}_{q \mapsto \mathsf{tte}}, i \models \psi'$. Now, consider a strategy $\mathsf{f} \in \mathrm{Str}(\mathtt{s}_0)$ such that $\mathsf{f}((\pi)_{\leq n}) = \mathsf{tte}(n)$, for all $n \in [i, \omega[$. Then, it is evident that $\chi^{\mathtt{x}_q \mapsto \mathsf{f}}(\mathtt{x}_{q'})((\pi)_{\leq n}) = \mathsf{pte}_{q \mapsto \mathsf{tte}}(q')(n)$, for all $q' \in \mathsf{free}(\psi)$ and $n \in [i, \omega[$. So, by the inductive hypothesis, it follows that $\mathcal{G}_{Rdc}, (\chi^{\mathtt{x}_q \mapsto \mathsf{f}}, \mathtt{s}_0)^i \models \overline{\psi'}$. Thus, we have that $\mathcal{G}_{Rdc}, (\chi, \mathtt{s}_0)^i \models \langle\!\langle \mathtt{x}_q \rangle\!\rangle \overline{\psi'}$.

  *[If].* If $\mathcal{G}_{Rdc}, (\chi, \mathtt{s}_0)^i \models \langle\!\langle \mathtt{x}_q \rangle\!\rangle \overline{\psi'}$, there exists a strategy $\mathsf{f} \in \mathrm{Str}(\mathtt{s}_0)$ such that $\mathcal{G}_{Rdc}, (\chi^{\mathtt{x}_q \mapsto \mathsf{f}}, \mathtt{s}_0)^i \models \overline{\psi'}$. Now, consider a temporal truth evaluation $\mathsf{tte} \in \mathrm{TTE}$ such that $\mathsf{tte}(n) = \mathsf{f}((\pi)_{\leq n})$, for all $n \in [i, \omega[$. Then, it is evident that $\chi^{\mathtt{x}_q \mapsto \mathsf{f}}(\mathtt{x}_{q'})((\pi)_{\leq n}) = \mathsf{pte}_{q \mapsto \mathsf{tte}}(q')(n)$, for all $q' \in \mathsf{free}(\psi)$ and $n \in [i, \omega[$. So, by the inductive hypothesis, it follows that $\mathsf{pte}_{q \mapsto \mathsf{tte}}, i \models \psi'$. Thus, by Item 3a of Definition 1.2.2 on page 16, we have that $\mathsf{pte}, i \models \exists q.\psi'$.

- $\psi = \forall q.\psi'$.

  *[Only if].* For each strategy $\mathsf{f} \in \mathrm{Str}(\mathtt{s}_0)$, consider a temporal truth evaluation $\mathsf{tte} \in \mathrm{TTE}$ such that $\mathsf{tte}(n) = \mathsf{f}((\pi)_{\leq n})$, for all $n \in [i, \omega[$. It is evident that $\chi^{\mathtt{x}_q \mapsto \mathsf{f}}(\mathtt{x}_{q'})((\pi)_{\leq n}) = \mathsf{pte}_{q \mapsto \mathsf{tte}}(q')(n)$, for all $q' \in \mathsf{free}(\psi)$ and $n \in [i, \omega[$. Now, since $\mathsf{pte}, i \models \forall q.\psi'$, by Item 3b of Definition 1.2.2 on page 16, it follows that $\mathsf{pte}_{q \mapsto \mathsf{tte}}, i \models \psi'$. So, by the inductive hypothesis, for each strategy $\mathsf{f} \in \mathrm{Str}(\mathtt{s}_0)$, it holds that $\mathcal{G}_{Rdc}, (\chi^{\mathtt{x}_q \mapsto \mathsf{f}}, \mathtt{s}_0)^i \models \overline{\psi'}$. Thus, we have that $\mathcal{G}_{Rdc}, (\chi, \mathtt{s}_0)^i \models [\![\mathtt{x}_q]\!] \overline{\psi'}$.

  *[If].* For each temporal truth evaluation $\mathsf{tte} \in \mathrm{TTE}$, consider a strategy $\mathsf{f} \in \mathrm{Str}(\mathtt{s}_0)$ such that $\mathsf{f}((\pi)_{\leq n}) = \mathsf{tte}(n)$, for all $n \in [i, \omega[$. It is evident that $\chi^{\mathtt{x}_q \mapsto \mathsf{f}}(\mathtt{x}_{q'})((\pi)_{\leq n}) = \mathsf{pte}_{q \mapsto \mathsf{tte}}(q')(n)$, for all $q' \in \mathsf{free}(\psi)$ and $n \in [i, \omega[$. Now, since $\mathcal{G}_{Rdc}, (\chi, \mathtt{s}_0)^i \models [\![\mathtt{x}_q]\!] \overline{\psi'}$, it follows that $\mathcal{G}_{Rdc}, (\chi^{\mathtt{x}_q \mapsto \mathsf{f}}, \mathtt{s}_0)^i \models \overline{\psi'}$. So, by the inductive hypothesis, for each temporal truth evaluation $\mathsf{tte} \in \mathrm{TTE}$, it holds that $\mathsf{pte}_{q \mapsto \mathsf{tte}}, i \models \psi'$. Thus, by Item 3b of Definition 1.2.2 on page 16, we have that $\mathsf{pte}, i \models \forall q.\psi'$.

Thus, we are done with the proof. □

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

**Theorem 1.2.1 (**SL **Model-Checking Hardness)** *The model-checking problem for* SL[$k$-ALT] *is* $k$-EXPSPACE-HARD.

*Proof.* Let $\varphi$ be a QPTL[$k$-ALT] sentence, $\overline{\varphi}$ the related SL[$k$-ALT] variable-closed formula, and $\mathcal{G}_{Rdc}$ the CGS of Lemma 1.2.1 on the previous page of QPTL reduction. Then, by applying the previous mentioned lemma, it is easy to see that $\varphi$ is satisfiable iff $\mathcal{G}_{Rdc} \models [\![\mathtt{x}]\!](\alpha, \mathtt{x})\overline{\varphi}$ iff $\mathcal{G}_{Rdc} \models \langle\!\langle \mathtt{x} \rangle\!\rangle(\alpha, \mathtt{x})\overline{\varphi}$. Thus, the satisfiability problem for QPTL can be reduced to the model-checking problem for SL. Now, since the satisfiability problem for QPTL[$k$-ALT] is $k$-EXPSPACE-HARD [SVW87], we have that the model-checking problem for SL[$k$-ALT] is $k$-EXPSPACE-HARD as well. □

## 1.3 Satisfiability

In Section 1.2 it has been shown that the model-checking for SL is NONELEMENTARY-HARD. Here, we prove that the satisfiability problem is even harder, *i.e.*, undecidable. To do this, we first introduce a sentence that is satisfiable only on unbounded models. Then, by using this result, we prove the undecidability result through a reduction of the classic domino problem [Wan61].

### 1.3.1 Unbounded models

We now show that SL does not enjoy the bounded-tree model property. Informally, a modal logic satisfies the bounded-tree model property if, whenever a formula is satisfiable, it is so on a model in which all states have the number of successors bounded by an a priori fixed constant. Clearly, if a logic invariant under unwinding enjoys the finite model property, it enjoys the bounded-tree model property as well. The other direction may not hold, instead.

To prove this result for SL, we introduce, in the following definition, the sentence $\varphi^{ord}$ to be used as a counterexample.

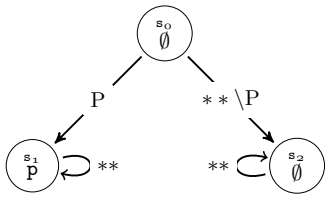



Figure 1.9: The CGS $\mathcal{G}^\star$ model of $\varphi^{ord}$.

**Definition 1.3.1 (Ordering Sentence)** *Let* $x_1 < x_2 \triangleq \langle\!\langle \mathtt{y} \rangle\!\rangle\ \varphi(x_1, x_2, \mathtt{y})$ *be an agent-closed formula, named* partial order*, on the sets* $\mathrm{AP} = \{\mathtt{p}\}$ *and* $\mathrm{Ag} = \{\alpha, \beta\}$*, where* $\varphi(x_1, x_2, \mathtt{y}) \triangleq ((\alpha, x_1)(\beta, \mathtt{y})(\mathtt{Xp})) \wedge ((\alpha, x_2)(\beta, \mathtt{y})(\mathtt{X}\neg\mathtt{p}))$*. Then, the* ordering sentence $\varphi^{ord} \triangleq \varphi^{unb} \wedge \varphi^{trn}$ *is the conjunction of the following two sentences, called* unboundedness *and* transitivity *strategy requirements:*

1. $\varphi^{unb} \triangleq [\![x_1]\!]\langle\!\langle x_2 \rangle\!\rangle\ x_1 < x_2$*;*
2. $\varphi^{trn} \triangleq [\![x_1]\!][\![x_2]\!][\![x_3]\!]\ (x_1 < x_2 \wedge x_2 < x_3) \rightarrow x_1 < x_3$.

Intuitively, $\varphi^{unb}$ asserts that, for each strategy in $x_1$, there is a different strategy in $x_2$ that is in relation of $<$ *w.r.t.* the first one, *i.e.*, $<$ has no upper bound, due to the fact that, by the definition of $\varphi(x_1, x_2, \mathtt{y})$, it is not reflexive. Moreover, $\varphi^{trn}$ ensures that the relation $<$ is transitive too. Consequently, $\varphi^{ord}$ induces a strict partial pre-order on the strategies.

Obviously, in order to be useful, the sentence $\varphi^{ord}$ needs to be satisfiable, as reported in the following lemma.

**Lemma 1.3.1 (Ordering Satisfiability)** *The sentence* $\varphi^{ord}$ *is satisfiable.*

*Proof.* To prove that $\varphi^{ord}$ is satisfiable, consider the unbounded CGS $\mathcal{G}^\star$, where *(i)* $\mathrm{AP} \triangleq \{p\}$, *(ii)* $\mathrm{Ag} \triangleq \{\alpha, \beta\}$, *(iii)* $\mathrm{Ac}_{\mathcal{G}^\star} \triangleq \mathbb{N}$, *(iv)* $\mathrm{St}_{\mathcal{G}^\star} \triangleq \{s_0, s_1, s_2\}$, *(v)* $s_{0\mathcal{G}^\star} = s_0$, *(vi)* $\mathsf{L}_{\mathcal{G}^\star}(s_0) = \mathsf{L}_{\mathcal{G}^\star}(s_2) \triangleq \emptyset$ and $\mathsf{L}_{\mathcal{G}^\star}(s_1) \triangleq \{p\}$, and *(vii)* $\mathsf{tr}_{\mathcal{G}^\star}$ is such that if $\delta \in \mathrm{P} \triangleq \{\delta \in \mathrm{Dc}_{\mathcal{G}^\star} : \delta(\alpha) \leq \delta(\beta)\}$ then $\mathsf{tr}_{\mathcal{G}^\star}(s_0, \delta) = s_1$ else $\mathsf{tr}_{\mathcal{G}^\star}(s_0, \delta) = s_2$, and $\mathsf{tr}_{\mathcal{G}^\star}(s, \delta) = s$, for all $s \in \{s_1, s_2\}$ and $\delta \in \mathrm{Dc}_{\mathcal{G}^\star}$ (see Figure 1.9).

Now, it is easy to see that $\mathcal{G}^\star \models \varphi^{unb}$, since for every strategy $\mathsf{f}_{x_1}$ for $x_1$, consisting of picking a natural number $n = \mathsf{f}_{x_1}(s_0)$ as an action at the initial state, we can reply with a strategy $\mathsf{f}_{x_2}$ for $x_2$ having $\mathsf{f}_{x_2}(s_0) > n$ and a strategy $\mathsf{f}_y$ for $y$ having $\mathsf{f}_y(s_0) = n$. Formally, we have that $\mathcal{G}^\star, \chi, s_0 \models \varphi(x_1, x_2, y)$ iff $\chi(x_1)(s_0) \leq \chi(y)(s_0) < \chi(x_2)(s_0)$, for all assignments $\chi \in \mathrm{Asg}_{\mathcal{G}^\star}(\{x_1, x_2, y\}, s_0)$.

By a similar reasoning, we can see that $\mathcal{G}^\star \models \varphi^{trn}$. Indeed, consider three strategies $\mathsf{f}_{x_1}$, $\mathsf{f}_{x_2}$, and $\mathsf{f}_{x_3}$ for the variables $x_1$, $x_2$, and $x_3$, respectively, which correspond to picking three natural numbers $n_1 = \mathsf{f}_{x_1}(s_0)$, $n_2 = \mathsf{f}_{x_2}(s_0)$, and $n_3 = \mathsf{f}_{x_3}(s_0)$. Now, if $\mathcal{G}^\star, \chi, s_0 \models x_1 < x_2$ and $\mathcal{G}^\star, \chi, s_0 \models x_2 < x_3$, for an assignments $\chi \in \mathrm{Asg}_{\mathcal{G}^\star}(\{x_1, x_2, x_3\}, s_0)$ where $\chi(x_1) = \mathsf{f}_{x_1}$, $\chi(x_2) = \mathsf{f}_{x_2}$, and $\chi(x_3) = \mathsf{f}_{x_3}$, we have that $n_1 < n_2$ and $n_2 < n_3$. Consequently, $n_1 < n_3$. Hence, by using a strategy $\mathsf{f}_y$ for $y$ with $\mathsf{f}_y(s_0) = \mathsf{f}_{x_1}(s_0)$, we have $\mathcal{G}^\star, \chi_{y \mapsto \mathsf{f}_y}, s_0 \models \varphi(x_1, x_3, y)$ and thus $\mathcal{G}^\star, \chi, s_0 \models x_1 < x_3$. □

Next lemmas report two important properties of the sentence $\varphi^{ord}$, for the negative statements we want to show. Namely, they state that, in order to be satisfied, $\varphi^{ord}$ must require the existence of strict partial order relations on strategies and actions that do not admit any maximal element. From this, as stated in Theorem 1.3.1 on the next page, we directly derive that $\varphi^{ord}$ needs an infinite chain of actions to be satisfied, *i.e.*, it cannot have a bounded model.

**Lemma 1.3.2 (Strategy Order)** *Let $\mathcal{G}$ be a model of $\varphi^{ord}$. Moreover, let $r^< \subseteq \mathrm{Str}_{\mathcal{G}}(s_{0\mathcal{G}}) \times \mathrm{Str}_{\mathcal{G}}(s_{0\mathcal{G}})$ be a relation between $s_{0\mathcal{G}}$-total strategies of $\mathcal{G}$ such that $r^<(\mathsf{f}_1, \mathsf{f}_2)$ holds iff $\mathcal{G}, \varnothing[x_1 \mapsto \mathsf{f}_1][x_2 \mapsto \mathsf{f}_2], s_{0\mathcal{G}} \models x_1 < x_2$, for all strategies $\mathsf{f}_1, \mathsf{f}_2 \in \mathrm{Str}_{\mathcal{G}}(s_{0\mathcal{G}})$. Then, $r^<$ is a* strict partial order without maximal element.

*Proof.* The proof derives from the fact that $r^<$ satisfies the following properties:

1. *Irreflexivity*: $\forall \mathsf{f} \in \mathrm{Str}_{\mathcal{G}}(s_{0\mathcal{G}}).\ \neg r^<(\mathsf{f}, \mathsf{f})$;
2. *Unboundedness*: $\forall \mathsf{f}_1 \in \mathrm{Str}_{\mathcal{G}}(s_{0\mathcal{G}})\ \exists \mathsf{f}_2 \in \mathrm{Str}_{\mathcal{G}}(s_{0\mathcal{G}}).\ r^<(\mathsf{f}_1, \mathsf{f}_2)$;
3. *Transitivity*: $\forall \mathsf{f}_1, \mathsf{f}_2, \mathsf{f}_3 \in \mathrm{Str}_{\mathcal{G}}(s_{0\mathcal{G}}).\ (r^<(\mathsf{f}_1, \mathsf{f}_2) \wedge r^<(\mathsf{f}_2, \mathsf{f}_3)) \rightarrow r^<(\mathsf{f}_1, \mathsf{f}_3)$.

Indeed, Items *(ii)* and *(iii)* are directly derived from the strategy unboundedness and transitivity requirements. The proof of Item *(i)* derives, instead, from the following reasoning. By contradiction, suppose that $r^<$ is not a strict order, *i.e.*, there is a strategy $\mathsf{f} \in \mathrm{Str}_{\mathcal{G}}(s_{0\mathcal{G}})$ for which $r^<(\mathsf{f}, \mathsf{f})$ holds. This means that, at the initial state $s_{0\mathcal{G}}$ of $\mathcal{G}$, there exists an assignment $\chi \in \mathrm{Asg}_{\mathcal{G}}(\{x_1, x_2, y\}, s_{0\mathcal{G}})$ for which $\mathcal{G}, \chi, s_{0\mathcal{G}} \models \varphi(x_1, x_2, y)$, where $\chi(x_1) = \chi(x_2) = \mathsf{f}$. The last fact implies the existence of a successor of $s_{0\mathcal{G}}$ in which both $p$ and $\neg p$ hold, which is clearly impossible. □

**Lemma 1.3.3 (Action Order)** *Let $\mathcal{G}$ be a model of $\varphi^{ord}$. Moreover, let $s^< \subseteq \mathrm{Ac}_{\mathcal{G}} \times \mathrm{Ac}_{\mathcal{G}}$ be a relation between actions of $\mathcal{G}$ such that $s^<(c_1, c_2)$ holds iff, for all strategies $\mathsf{f}_1, \mathsf{f}_2 \in \mathrm{Str}_{\mathcal{G}}(s_{0\mathcal{G}})$ with $c_1 = \mathsf{f}_1(s_{0\mathcal{G}})$ and $c_2 = \mathsf{f}_2(s_{0\mathcal{G}})$, it holds that $r^<(\mathsf{f}_1, \mathsf{f}_2)$, where $c_1, c_2 \in \mathrm{Ac}_{\mathcal{G}}$. Then, $s^<$ is a* strict partial order without maximal element.

*Proof.* The irreflexivity and transitivity of $s^<$ are directly derived from the fact that, by Lemma 1.3.2 on the facing page, $r^<$ is irreflexive and transitive too. The proof of the unboundedness property derives, instead, from the following reasoning. As first thing, observe that, since the formula $x_1 < x_2$ relies on $\mathtt{X}p$ and $\mathtt{X}\neg p$ as the only temporal operators, it holds that $r^<(\mathsf{f}_1, \mathsf{f}_2)$ implies $r^<(\mathsf{f}'_1, \mathsf{f}'_2)$, for all strategies $\mathsf{f}_1, \mathsf{f}_2, \mathsf{f}'_1, \mathsf{f}'_2 \in \mathrm{Str}_{\mathcal{G}}(s_{0\mathcal{G}})$ such that $\mathsf{f}_1(s_{0\mathcal{G}}) = \mathsf{f}'_1(s_{0\mathcal{G}})$ and $\mathsf{f}_2(s_{0\mathcal{G}}) = \mathsf{f}'_2(s_{0\mathcal{G}})$. Now, suppose by contradiction that $s^<$ does not satisfy the unboundedness property, *i.e.*, there is an action $c \in \mathrm{Ac}_{\mathcal{G}}$ such that, for all actions $c' \in \mathrm{Ac}_{\mathcal{G}}$, it results that $s^<(c, c')$ does not hold. Then, by the definition of $s^<$ and the previous observation, we derive the existence of a strategy $\mathsf{f} \in \mathrm{Str}_{\mathcal{G}}(s_{0\mathcal{G}})$ with $\mathsf{f}(s_{0\mathcal{G}}) = c$ such that $r^<(\mathsf{f}, \mathsf{f}')$ does not hold, for any strategy $\mathsf{f}' \in \mathrm{Str}_{\mathcal{G}}(s_{0\mathcal{G}})$, which is clearly impossible. □

Now, we have all tools to prove that SL lacks of the bounded-tree model property, which hold, instead, for several commonly used multi-agent logics, such as ATL$^\star$.

**Theorem 1.3.1 (SL Unbounded Model Property)** SL *does not enjoy the bounded model property.*

*Proof.* To prove the statement, we show that the sentence $\varphi^{ord}$ of Definition 1.3.1 on page 19 cannot be satisfied on a bounded CGS. Consider a CGS $\mathcal{G}$ such that $\mathcal{G} \models \varphi^{ord}$. The existence of such a model is ensured by Lemma 1.3.1 on page 19. Now, consider the strict partial order without maximal element between actions $s^<$ described in Lemma 1.3.3 on the preceding page. By a classical result on first order logic model theory [EF95], the relation $s^<$ cannot be defined on a finite set. Hence, $|\mathrm{Ac}| = \infty$.

□

### 1.3.2 Undecidable satisfiability

We finally show the undecidability of the satisfiability problem for SL through a reduction from the *recurrent domino problem*.

The *domino problem*, proposed for the first time by Wang [Wan61], consists of placing a given number of tile types on an infinite grid, satisfying a predetermined set of constraints on adjacent tiles. One of its standard versions asks for a compatible tiling of the whole plane $\mathbb{N} \times \mathbb{N}$. The *recurrent domino problem* further requires the existence of a distinguished tile type that occurs infinitely often in the first row of the grid. This problem was proved to be highly undecidable by Harel, and in particular, $\Sigma^1_1$-COMPLETE [Har84]. The formal definition follows.

**Definition 1.3.2 (Recurrent Domino System)** *An* $\mathbb{N} \times \mathbb{N}$ recurrent domino system $\mathcal{D} = \langle \mathrm{D}, H, V, t_0 \rangle$ *consists of a finite non-empty set* D *of* domino types*, two* horizontal *and* vertical matching relations $H, V \subseteq \mathrm{D} \times \mathrm{D}$*, and a* distinguished tile type $t_0 \in \mathrm{D}$*. The recurrent domino problem asks for an* admissible tiling *of* $\mathbb{N} \times \mathbb{N}$*, which is a* solution mapping $\partial : \mathbb{N} \times \mathbb{N} \to \mathrm{D}$ *such that, for all* $x, y \in \mathbb{N}$*, it holds that* (i) $(\partial(x, y), \partial(x + 1, y)) \in H$*,* (ii) $(\partial(x, y), \partial(x, y + 1)) \in V$*, and* (iii) $|\{x \in \mathbb{N} : \partial(x, 0) = t_0\}| = \omega$.

### 1.3. Satisfiability

**Grid specification** Consider the sentence $\varphi^{grd} \triangleq \bigwedge_{a\in \mathrm{Ag}} \varphi_a^{ord}$, where $\varphi_a^{ord} = \varphi_a^{unb} \wedge \varphi_a^{trn}$ are the *order sentences* and $\varphi_a^{unb}$ and $\varphi_a^{trn}$ are the *unboundedness* and *transitivity* strategy requirements for agents $\alpha$ and $\beta$ defined, similarly to Definition 1.3.1 on page 19, as follows:

1. $\varphi_a^{unb} \triangleq [\![z_1]\!]\langle\!\langle z_2\rangle\!\rangle\, z_1 <_a z_2$;
2. $\varphi_a^{trn} \triangleq [\![z_1]\!][\![z_2]\!][\![z_3]\!]\, (z_1 <_a z_2 \wedge z_2 <_a z_3) \rightarrow z_1 <_a z_3$;

where $x_1 <_\alpha x_2 \triangleq \langle\!\langle y\rangle\!\rangle \varphi_\alpha(x_1, x_2, y)$ and $y_1 <_\beta y_2 \triangleq \langle\!\langle x\rangle\!\rangle \varphi_\beta(y_1, y_2, x)$ are the two *partial order* formulas on strategies of $\alpha$ and $\beta$, respectively, with $\varphi_\alpha(x_1, x_2, y) \triangleq (\beta, y)((\alpha, x_1)(\mathtt{X}p) \wedge (\alpha, x_2)(\mathtt{X}\neg p))$ and $\varphi_\beta(y_1, y_2, x) \triangleq (\alpha, x)((\beta, y_1)(\mathtt{X}\neg p) \wedge (\beta, y_2)(\mathtt{X}p))$. Intuitively, $<_\alpha$ and $<_\beta$ correspond to the horizontal and vertical ordering of the positions in the grid, respectively.

It is easy to show that $\varphi^{grd}$ is satisfiable, by using the same candidate model $\mathcal{G}^\star$ (see Figure 1.9 on page 19)and a proof argument similar to that proposed in Lemma 1.3.1 on page 19 for the simpler order sentence.

**Lemma 1.3.4 (Grid Ordering Satisfiability)** *The sentence $\varphi^{grd}$ is satisfiable.*

*Proof.* Let $\mathcal{G}^\star$ be the model described in Lemma 1.3.1 on page 19. On one hand, by using the same lemma, it is evident that $\mathcal{G}^\star \models \varphi_\alpha^{ord}$. On the other hand, in order to prove that $\mathcal{G}^\star \models \varphi_\beta^{ord}$, first observe that $\mathcal{G}^\star, \chi, s_{0\mathcal{G}^\star} \models \varphi_\beta(y_1, y_2, x)$ iff $\chi(y_1)(s_{0\mathcal{G}^\star}) < \chi(x)(s_{0\mathcal{G}^\star}) \leq \chi(y_2)(s_{0\mathcal{G}^\star})$, for all assignments $\chi \in \mathrm{Asg}_{\mathcal{G}^\star}(\{y_1, y_2, x\}, s_{0\mathcal{G}^\star})$. At this point, the thesis follows by a reasoning similar to the one proposed in the lemma. □

Consider now a model $\mathcal{G}$ of $\varphi^{grd}$ and, for all agents $a \in \mathrm{Ag}$, the relation $r_a^< \subseteq \mathrm{Str}_\mathcal{G}(s_{0\mathcal{G}}) \times \mathrm{Str}_\mathcal{G}(s_{0\mathcal{G}})$ between $s_{0\mathcal{G}}$-total strategies defined as follows: $r_a^<(\mathsf{f}_1, \mathsf{f}_2)$ holds iff $\mathcal{G}, \varnothing[z_1 \mapsto \mathsf{f}_1][z_2 \mapsto \mathsf{f}_2], s_{0\mathcal{G}} \models z_1 <_a z_2$, for all strategies $\mathsf{f}_1, \mathsf{f}_2 \in \mathrm{Str}_\mathcal{G}(s_{0\mathcal{G}})$. By using a proof similar to that of Lemma 1.3.2 on page 20, it is possible to see that $r_a^<$ is a *strict partial order without maximal element* on $\mathrm{Str}_\mathcal{G}(s_{0\mathcal{G}})$.

Now, to apply the desired reduction, we need to transform $r_a^<$ into a total order over strategies, by using the following two lemmas.

**Lemma 1.3.5 (Strategy Equivalence)** *Let $r_a^\equiv \subseteq \mathrm{Str}_\mathcal{G}(s_{0\mathcal{G}}) \times \mathrm{Str}_\mathcal{G}(s_{0\mathcal{G}})$, with $a \in \mathrm{Ag}$, be the relation between strategies such that $r_a^\equiv(\mathsf{f}_1, \mathsf{f}_2)$ holds iff neither $r_a^<(\mathsf{f}_1, \mathsf{f}_2)$ nor $r_a^<(\mathsf{f}_2, \mathsf{f}_1)$ holds, for all $\mathsf{f}_1, \mathsf{f}_2 \in \mathrm{Str}_\mathcal{G}(s_{0\mathcal{G}})$. Then $r_a^\equiv$ is an* equivalence relation.

*Proof.* It is immediate to see that the relation $r_a^\equiv$ is reflexive, since $r_a^<$ is not reflexive. Moreover, it is symmetric by definition. Finally, due to the definition of the partial order formula $<_a$, it is also transitive and, thus, $r_a^\equiv$ is an *equivalence relation*. Indeed, if both $r_\alpha^\equiv(\mathsf{f}_1, \mathsf{f}_2)$ and $r_\alpha^\equiv(\mathsf{f}_2, \mathsf{f}_3)$ hold, we have that either $\mathcal{G}, \chi, s_{0\mathcal{G}} \models (\beta, y)((\alpha, x_1)(\mathtt{X}p) \wedge (\alpha, x_2)(\mathtt{X}p))$ or $\mathcal{G}, \chi, s_{0\mathcal{G}} \models (\beta, y)((\alpha, x_1)(\mathtt{X}\neg p) \wedge (\alpha, x_2)(\mathtt{X}\neg p))$ and either $\mathcal{G}, \chi, s_{0\mathcal{G}} \models (\beta, y)((\alpha, x_2)(\mathtt{X}p) \wedge (\alpha, x_3)(\mathtt{X}p))$ or $\mathcal{G}, \chi, s_{0\mathcal{G}} \models (\beta, y)((\alpha, x_2)(\mathtt{X}\neg p) \wedge (\alpha, x_3)(\mathtt{X}\neg p))$, for all assignments $\chi \in \mathrm{Asg}(\mathcal{G}, \{x_1, x_2, x_3, y\}, s_{0\mathcal{G}})$ such that $\chi(x_1) = \mathsf{f}_1$, $\chi(x_2) = \mathsf{f}_2$, and $\chi(x_3) = \mathsf{f}_3$. Hence, we also have that either $\mathcal{G}, \chi, s_{0\mathcal{G}} \models (\beta, y)((\alpha, x_1)(\mathtt{X}p) \wedge (\alpha, x_3)(\mathtt{X}p))$ or $\mathcal{G}, \chi, s_{0\mathcal{G}} \models (\beta, y)((\alpha, x_1)(\mathtt{X}\neg p) \wedge (\alpha, x_3)(\mathtt{X}\neg p))$. Thus, $r_\alpha^\equiv(\mathsf{f}_1, \mathsf{f}_3)$ holds, too. The same reasoning applies to $r_\beta^\equiv$. □

Let $\mathrm{Str}_a^\equiv \triangleq (\mathrm{Str}_\mathcal{G}(s_{0\mathcal{G}})/r_a^\equiv)$ be the quotient set of $\mathrm{Str}_\mathcal{G}(s_{0\mathcal{G}})$ *w.r.t.* $r_a^\equiv$, for $a \in \mathrm{Ag}$, *i.e.*, the set of the related equivalence classes over $s_{0\mathcal{G}}$-total strategies. Then, the following holds.

**Lemma 1.3.6 (Strategy Total Order)** *Let $s_a^< \subseteq \mathrm{Str}_a^{\equiv} \times \mathrm{Str}_a^{\equiv}$, with $a \in \mathrm{Ag}$, be the relation between classes of strategies such that $s_a^<(\mathrm{F}_1, \mathrm{F}_2)$ holds iff $r_a^<(\mathsf{f}_1, \mathsf{f}_2)$ holds, for all $\mathsf{f}_1 \in \mathrm{F}_1$, $\mathsf{f}_2 \in \mathrm{F}_2$, and $\mathrm{F}_1, \mathrm{F}_2 \in \mathrm{Str}_a^{\equiv}$. Then $s_a^<$ is a* strict total order with minimal element but no maximal element.

*Proof.* The fact that $s_a^<$ is a *strict partial order without maximal element* derives directly from the same property of $r_a^<$. Indeed, due to the definition of the partial order formula $<_a$, if $r_a^{\equiv}(\mathsf{f}', \mathsf{f}'')$ and $r_a^<(\mathsf{f}', \mathsf{f})$ (resp., $r_a^<(\mathsf{f}, \mathsf{f}')$) hold, we obtain that $r_a^<(\mathsf{f}'', \mathsf{f})$ (resp., $r_a^<(\mathsf{f}, \mathsf{f}'')$) holds too. Hence, if there are $\mathsf{f}_1 \in \mathrm{F}_1$ and $\mathsf{f}_2 \in \mathrm{F}_2$ such that $r_a^<(\mathsf{f}_1, \mathsf{f}_2)$ holds, we directly obtain that $s_a^<(\mathrm{F}_1, \mathrm{F}_2)$ holds as well, for all $\mathrm{F}_1, \mathrm{F}_2 \in \mathrm{Str}_a^{\equiv}$ and $a \in \mathrm{Ag}$.

Moreover, $s_a^<$ is total, since $r_a^{\equiv}$ is an equivalence relation that cluster together all strategies of the agent $a$ that are not in relation *w.r.t.* either $r_a^<$ or its inverse $(r_a^<)^{-1}$. Indeed, suppose by contradiction that there are two different classes $\mathrm{F}_1, \mathrm{F}_2 \in \mathrm{Str}_a^{\equiv}$ such that neither $s_a^<(\mathrm{F}_1, \mathrm{F}_2)$ nor $s_a^<(\mathrm{F}_2, \mathrm{F}_1)$ holds. This means that, for all $\mathsf{f}_1 \in \mathrm{F}_1$ and $\mathsf{f}_2 \in \mathrm{F}_2$, neither $r_a^<(\mathsf{f}_1, \mathsf{f}_2)$ nor $r_a^<(\mathsf{f}_2, \mathsf{f}_1)$ holds and, so, $r_a^{\equiv}(\mathsf{f}_1, \mathsf{f}_2)$. But, this contradicts the fact that $\mathrm{F}_1$ and $\mathrm{F}_2$ are different equivalences classes.

Finally, it is important to note that in $\mathrm{Str}_a^{\equiv}$ there is also a minimal element *w.r.t.* $s_a^<$. Indeed, for a strategy $\mathsf{f} \in \mathrm{Str}_{\mathcal{G}}(s_{0\mathcal{G}})$ for $\alpha$ (resp., for $\beta$) that forces the play to reach only nodes labeled with $p$ (resp., $\neg p$) as successor of $s_{0\mathcal{G}}$, independently from the strategy of $\beta$ (resp., $\alpha$), the relation $r_\alpha^<(\mathsf{f}', \mathsf{f})$ (resp., $r_\beta^<(\mathsf{f}', \mathsf{f})$) cannot hold, for any $\mathsf{f}' \in \mathrm{Str}_{\mathcal{G}}(s_{0\mathcal{G}})$. □

By a classical result on first-order model theory [EF95], the relation $s_a^<$ cannot be defined on a finite set. Hence, $|\mathrm{Str}_a^{\equiv}| = \omega$, for all $a \in \mathrm{Ag}$. Now, let $s_a^{\prec} \subseteq \mathrm{Str}_a^{\equiv} \times \mathrm{Str}_a^{\equiv}$ be the *successor* relation on $\mathrm{Str}_a^{\equiv}$ compatible with the strict total order $s_a^<$, *i.e.*, such that $s_a^{\prec}(\mathrm{F}_1, \mathrm{F}_2)$ holds iff *(i)* $s_a^<(\mathrm{F}_1, \mathrm{F}_2)$ holds and *(ii)* there is no $\mathrm{F}_3 \in \mathrm{Str}_a^{\equiv}$ for which both $s_a^<(\mathrm{F}_1, \mathrm{F}_3)$ and $s_a^<(\mathrm{F}_3, \mathrm{F}_2)$ hold, for all $\mathrm{F}_1, \mathrm{F}_2 \in \mathrm{Str}_a^{\equiv}$. Then, we can represent the two sets of classes $\mathrm{Str}_\alpha^{\equiv}$ and $\mathrm{Str}_\beta^{\equiv}$, respectively, as the infinite ordered lists $\{\mathrm{F}_0^\alpha, \mathrm{F}_1^\alpha, \ldots\}$ and $\{\mathrm{F}_0^\beta, \mathrm{F}_1^\beta, \ldots\}$ such that $s_a^{\prec}(\mathrm{F}_i^a, \mathrm{F}_{i+1}^a)$ holds, for all indexes $i \in \mathbb{N}$. Note that $\mathrm{F}_0^a$ is the class of minimal strategies w.r.t the relation $s_a^<$.

At this point, we have all the machinery to build an embedding of the plane $\mathbb{N} \times \mathbb{N}$ into a model $\mathcal{G}$ of $\varphi^{grd}$. Formally, we consider the *bijective map* $\aleph : \mathbb{N} \times \mathbb{N} \to \mathrm{Str}_\alpha^{\equiv} \times \mathrm{Str}_\beta^{\equiv}$ such that $\aleph(i, j) = (\mathrm{F}_i^\alpha, \mathrm{F}_j^\beta)$, for all $i, j \in \mathbb{N}$.

**Compatible tiling** Given the grid structure built on the model $\mathcal{G}$ of $\varphi^{grd}$ through the bijective map $\aleph$, we can express that a tiling of the grid is admissible by making use of the formula $z_1 \prec_a z_2 \triangleq z_1 <_a z_2 \wedge \neg\langle\!\langle z_3 \rangle\!\rangle\, z_1 <_a z_3 \wedge z_3 <_a z_2$ corresponding to the successor relation $s_a^{\prec}$, for all $a \in \mathrm{Ag}$. Indeed, it is not hard to see that $\mathcal{G}, \chi, s_{0\mathcal{G}} \models z_1 \prec_a z_2$ iff $\chi(z_1) \in \mathrm{F}_i^a$ and $\chi(z_2) \in \mathrm{F}_{i+1}^a$, for all indexes $i \in \mathbb{N}$ and assignments $\chi \in \mathrm{Asg}_{\mathcal{G}}(\{z_1, z_2\}, s_{0\mathcal{G}})$. The idea here is to associate with each domino type $t \in \mathrm{D}$ a corresponding atomic proposition $t \in \mathrm{AP}$ and to express the horizontal and vertical matching conditions via suitable object labeling. In particular, we can express that the tiling is locally compatible, the horizontal neighborhoods of a tile satisfy the $H$ or $V$ requirements, respectively. All these constraints can be formulated through the following three agent-closed formulas:

1. $\varphi^{t,loc}(x, y) \triangleq (\alpha, x)(\beta, y)(\mathtt{X}(t \wedge \bigwedge_{t' \in \mathrm{D}}^{t' \neq t} \neg t'))$;

2. $\varphi^{t,hor}(x,y) \triangleq \bigvee_{(t,t')\in H} [\![x']\!](x \prec_\alpha x' \rightarrow (\alpha,x')(\beta,y)(\mathtt{X}t'))$;

3. $\varphi^{t,ver}(x,y) \triangleq \bigvee_{(t,t')\in V} [\![y']\!](y \prec_\beta y' \rightarrow (\alpha,x)(\beta,y')(\mathtt{X}t'))$.

Informally, we have the following: $\varphi^{t,loc}(x,y)$ asserts that $t$ is the only domino type labeling the successors of the root of the model $\mathcal{G}$ that can be reached using the strategies related to the variables $x$ and $y$; $\varphi^{t,hor}(x,y)$ asserts that the tile $t'$ labeling the successors of the root reachable through the strategies $x'$ and $y$ is compatible with $t$ *w.r.t.* the horizontal requirement $H$, for all strategies $x'$ that immediately follow that related to $x$ *w.r.t.* the order $r_\alpha^<$; $\varphi^{t,ver}(x,y)$ asserts that the tile $t'$ labeling the successors of the root reachable through the strategies $x$ and $y'$ is compatible with $t$ *w.r.t.* the vertical requirement $V$, for all strategies $y'$ that immediately follow that related to $y$ *w.r.t.* the order $r_\beta^<$.

Finally, to express that the whole grid has an admissible tiling, we use the sentence $\varphi^{til} \triangleq [\![x]\!][\![y]\!] \bigvee_{t\in \mathrm{D}} \varphi^{t,loc}(x,y) \wedge \varphi^{t,hor}(x,y) \wedge \varphi^{t,ver}(x,y)$ that asserts the existence of a domino type $t$ satisfying the three conditions mentioned above, for every point identified by the strategies $x$ and $y$.

**Recurrent tile** As last task, we impose that the grid embedded into $\mathcal{G}$ has the distinguished domino type $t_0$ occurring infinitely often in its first row. To do this, we describe two formulas that determine if a row or a column is the first one *w.r.t.* the orders $s_\alpha^<$ and $s_\beta^<$, respectively. Formally, we use $0_a(z) \triangleq \neg\langle\!\langle z'\rangle\!\rangle\, z' <_a z$, for $a \in \mathrm{Ag}$. One can easily prove that $\mathcal{G}, \chi, s_{0\mathcal{G}} \models 0_\alpha(z)$ iff $\chi(z) \in \mathrm{F}_0^a$, for all assignments $\chi \in \mathrm{Asg}_{\mathcal{G}}(\{z\}, s_{0\mathcal{G}})$. Now, the infinite occurrence requirement on $t_0$ can be expressed with the following sentence: $\varphi^{rec} \triangleq [\![x]\!][\![y]\!](0_\beta(y) \wedge (0_\alpha(x) \vee (\alpha,x)(\beta,y)(\mathtt{X}t_0))) \rightarrow \langle\!\langle x'\rangle\!\rangle\, x <_\alpha x' \wedge (\alpha,x')(\beta,y)(\mathtt{X}t_0)$. Informally, $\varphi^{rec}$ asserts that, when we are on the first row identified by the variable $y$ and at a column pointed by $x$ such that it is the first column or the node of the *"intersection"* between $x$ and $y$ is labeled by $t_0$, we have that there exists a greater column identified by $x'$ such that its *"intersection"* with $y$ is labeled by $t_0$ as well.

**Construction correctness** At this point, we have all tools to formally prove the correctness of the undecidability reduction, by showing the equivalence between the satisfiability of the sentence $\varphi^{dom}$ and finding a solution of the recurrent tiling problem.

**Theorem 1.3.2 (Satisfiability)** *The satisfiability problem for* SL *is highly undecidable. In particular, it is* $\Sigma_1^1$-HARD.

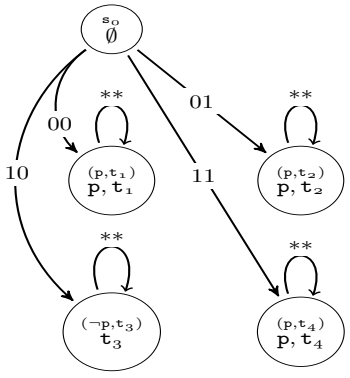



Figure 1.10: Part of the CGS $\mathcal{G}_\partial{}^\star$ model of $\varphi^{dom}$, where $\partial(0,0) = \mathtt{t}_1$, $\partial(0,1) = \mathtt{t}_2$, $\partial(1,0) = \mathtt{t}_3$, and $\partial(1,1) = \mathtt{t}_4$.

*Proof.* For the direct reduction, assume that there exists a solution mapping $\partial : \mathbb{N}\times\mathbb{N} \rightarrow \mathrm{D}$ for the given recurrent domino system $\mathcal{D}$. Then, we can build a finite CGS $\mathcal{G}_\partial^\star$ similar to the one used in Lemma 1.3.1 on page 19, which satisfies the sentence $\varphi^{dom}$:

(i) $\mathrm{Ac}_{\mathcal{G}^\star_\partial} \triangleq \mathbb{N}$;

(ii) there are $2 \cdot |\mathrm{D}| + 1$ different states $\mathrm{St}_{\mathcal{G}^\star_\partial} \triangleq \{s_0\} \cup (\{p, \neg p\} \times \mathrm{D})$ such that $\mathsf{L}_{\mathcal{G}^\star_\partial}(s_0) \triangleq \emptyset$, $\mathsf{L}_{\mathcal{G}^\star_\partial}((p,t)) \triangleq \{p,t\}$, and $\mathsf{L}_{\mathcal{G}^\star_\partial}((\neg p,t)) \triangleq \{t\}$, for all $t \in \mathrm{D}$;

(iii) each state $(z,t) \in \{p, \neg p\} \times \mathrm{D}$ has only self loops $\mathsf{tr}_{\mathcal{G}^\star_\partial}((z,t),\delta) \triangleq (z,t)$ and the initial state $s_{0\mathcal{G}^\star_\partial} \triangleq s_0$ is connected to $(z,t)$ through the decision $\delta$, *i.e.*, $\mathsf{tr}_{\mathcal{G}^\star_\partial}(s_0,\delta) \triangleq (z,t)$, iff

(a) $t = \partial(\delta(\alpha), \delta(\beta))$ and

(b) $z = p$ iff $\delta(\alpha) \leq \delta(\beta)$, for all $\delta \in \mathrm{Dc}_{\mathcal{G}^\star_\partial}$.

By a simple case analysis on the subformulas of $\varphi^{dom}$, it is possible to see that $\mathcal{G}^\star_\partial \models \varphi^{dom}$.

Conversely, let $\mathcal{G}$ be a model of the sentence $\varphi^{dom}$ and $\aleph : \mathbb{N} \times \mathbb{N} \to \mathrm{Str}^{\equiv}_\alpha \times \mathrm{Str}^{\equiv}_\beta$ the related bijective map built for the grid specification task. As first thing, we have to prove the existence of a coloring function $\eth : \mathrm{Str}^{\equiv}_\alpha \times \mathrm{Str}^{\equiv}_\beta \to \mathrm{D}$ such that, for all pairs of classes of strategies $(\mathrm{F}^\alpha, \mathrm{F}^\beta) \in \mathrm{Str}^{\equiv}_\alpha \times \mathrm{Str}^{\equiv}_\beta$ and assignments $\chi \in \mathrm{Asg}_\mathcal{G}(\{\alpha,\beta\}, s_{0\mathcal{G}})$ with $\chi(\alpha) \in \mathrm{F}^\alpha$ and $\chi(\beta) \in \mathrm{F}^\beta$, it holds that $\mathcal{G}, \chi, s_{0\mathcal{G}} \models \mathtt{X}\eth(\mathrm{F}^\alpha, \mathrm{F}^\beta)$. Then, it remains to note that the solution mapping $\partial = \eth \circ \aleph$ built as a composition of the bijective map $\aleph$ and the coloring function $\eth$ is an admissible tiling of the plane $\mathbb{N} \times \mathbb{N}$.

Due to the $\varphi^{t,loc}$ formula in the sentence $\varphi^{til}$, we have that, for all assignments $\chi \in \mathrm{Asg}_\mathcal{G}(\{\alpha,\beta\}, s_{0\mathcal{G}})$, there exists just one domino type $t \in \mathrm{D}$ satisfying the property $\mathcal{G}, \chi, s_{0\mathcal{G}} \models \mathtt{X}t$. Let $\widehat{\eth} : \mathrm{Str}_\mathcal{G}(s_{0\mathcal{G}}) \times \mathrm{Str}_\mathcal{G}(s_{0\mathcal{G}}) \to \mathrm{D}$ be the function that returns such a type, for all pairs of strategies of $\alpha$ and $\beta$, *i.e.*, such that $\mathcal{G}, \chi, s_{0\mathcal{G}} \models \mathtt{X}\widehat{\eth}(\chi(\alpha), \chi(\beta))$, for all assignments $\chi \in \mathrm{Asg}_\mathcal{G}(\{\alpha,\beta\}, s_{0\mathcal{G}})$. Now, it is not hard to see that, due to the formulas $\varphi^{t,hor}$ and $\varphi^{t,ver}$ in the sentence $\varphi^{til}$, it holds *(i)* $(\widehat{\eth}(\mathsf{f}_\alpha, \mathsf{f}_\beta), \widehat{\eth}(\mathsf{f}'_\alpha, \mathsf{f}_\beta)) \in H$ and *(ii)* $(\widehat{\eth}(\mathsf{f}_\alpha, \mathsf{f}_\beta), \widehat{\eth}(\mathsf{f}_\alpha, \mathsf{f}'_\beta)) \in V$, for all $\mathsf{f}_\alpha \in \mathrm{F}^\alpha_i$, $\mathsf{f}'_\alpha \in \mathrm{F}^\alpha_{i+1}$, $\mathsf{f}_\beta \in \mathrm{F}^\beta_j$, $\mathsf{f}'_\beta \in \mathrm{F}_{j+1}{}^\beta$, and $i,j \in \mathbb{N}$. Moreover, the guess of the tile type $t'$ adjacent to $t$ is uniform *w.r.t.* the choice of the successor strategy. Indeed, the disjunctions $\bigvee_{(t,t') \in H}$ and $\bigvee_{(t,t') \in V}$ precede the universal quantifications $[\![x']\!]$ and $[\![y']\!]$ in the formulas $\varphi^{t,hor}$ and $\varphi^{t,ver}$, respectively. Thus, we have that, for all $\mathsf{f}'_\alpha, \mathsf{f}''_\alpha \in \mathrm{F}^\alpha_i$ and $\mathsf{f}'_\beta, \mathsf{f}''_\beta \in \mathrm{F}^\beta_j$ with $i,j \in \mathbb{N}$ and $i + j > 0$, it holds that $\widehat{\eth}(\mathsf{f}'_\alpha, \mathsf{f}'_\beta) = \widehat{\eth}(\mathsf{f}''_\alpha, \mathsf{f}''_\beta)$. Note that this fact is not necessarily true for strategies belonging to the minimal classes $\mathrm{F}^\alpha_0$ and $\mathrm{F}^\beta_0$, since the sentence $\varphi^{dom}$ does not contain a relative requirement. However, every domino type $\widehat{\eth}(\mathsf{f}_\alpha, \mathsf{f}_\beta)$, with $\mathsf{f}_\alpha \in \mathrm{F}^\alpha_0$ and $\mathsf{f}_\beta \in \mathrm{F}^\beta_0$, can be used to label the origin of the plane $\mathbb{N} \times \mathbb{N}$ in order to obtain an admissible tiling. So, we can consider a function $\eth$, defined as follows: *(i)* $\eth(\mathrm{F}^\alpha_0, \mathrm{F}^\beta_0) \in \{\widehat{\eth}(\mathsf{f}_\alpha, \mathsf{f}_\beta) : \mathsf{f}_\alpha \in \mathrm{F}^\alpha_0 \wedge \mathsf{f}_\beta \in \mathrm{F}^\beta_0\}$; *(ii)* $\eth(\mathrm{F}^\alpha_i, \mathrm{F}^\beta_j) = \widehat{\eth}(\mathsf{f}_\alpha, \mathsf{f}_\beta)$, for all $\mathsf{f}_\alpha \in \mathrm{F}^\alpha_i$, $\mathsf{f}_\beta \in \mathrm{F}^\beta_j$, and $i,j \in \mathbb{N}$ with $i + j > 0$.

Clearly, *(i)* $(\eth(\mathrm{F}^\alpha_i, \mathrm{F}^\beta_j), \eth(\mathrm{F}^\alpha_{i+1}, \mathrm{F}^\beta_j)) \in H$, *(ii)* $(\eth(\mathrm{F}^\alpha_i, \mathrm{F}^\beta_j), \eth(\mathrm{F}^\alpha_i, \mathrm{F}_{j+1}{}^\beta)) \in V$, and *(iii)* $|\{i : \eth(\mathrm{F}^\alpha_i, \mathrm{F}^\beta_0) = t_0\}| = \omega$, for all $i,j \in \mathbb{N}$. So, $\partial = \eth \circ \aleph$ is an admissible tiling. $\square$

CHAPTER 2

# Strategy Logic Fragments

Since model checking for SL is non-elementary hard and satisfiability is even undecidable, while the same problems for ATL$^\star$ are only 2EXPTIME-COMPLETE, a question that arises naturally is whether there are proper fragments of SL of practical interest strictly subsuming ATL$^\star$ yet still having the same elementary complexities.

In this chapter, we answer positively to this question and go even further. More precisely, we reveal a fundamental property that, if satisfied, allows to retain a 2EXPTIME-COMPLETE model-checking problem. This is the property of *behavioral* quantification. Standard quantification over strategies introduces a somewhat unintuitive interpretation in the semantics of games, since it instantiates full strategies, which includes also future actions, whereas players see only actions played in the past. In more details, quantification of the form $\forall\exists$ means that the existentially quantified strategy is chosen in response to the universally quantified strategy, which prescribes actions in response to histories that may never arise in the actual game. On the other hand, in classic games player reacts only to current and past actions of other players, so the response of the existentially quantified strategy to a given history should depend only on the response of the universally quantified strategy to this history. To formally introduce the concept of behavioral quantification, we use the notion of *Skolem dependence function* as a machinery (see also [Kol85] for more on this subject). Furthermore, we show how the behavioral property can be used to apply a 2EXPTIME procedure to solve the model-checking problem. Finally, we prove the bounded model property, which is fundamental to provide a decidable procedure for the satisfiability problem.

The chapter is organized as follows. In Section 2.1, we describe three syntactic fragments of SL, named SL[NG], SL[BG], and SL[1G], having the peculiarity to use strategy quantifications grouped in atomic blocks. Then, in Section 2.2, we define the notion of Skolem dependence function, which is used, to introduce the concept of behavioral and then the prove a fundamental result, which is at the base of our elementary model-checking procedure for SL[1G]. In Section 2.3, we provide an automata-based 2EXPTIME procedure to solve the model-checking problem for SL[1G]. Finally, in Section 2.4 we first prove the *bounded-tree model property* for SL[1G] and then, by means of this, we provide a 2EXPTIME procedure for the satisfiability problem of this fragment.

## 2.1 Syntax

In order to formalize the syntactic fragments of SL we want to investigate, we first need to define the concepts of *quantification* and *binding prefixes*. A *quantification prefix* over a set $\mathrm{V} \subseteq \mathrm{Vr}$ 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 $\wp$, i.e., there is exactly one index $i \in [0, |\mathrm{V}|[$ such that $(\wp)_i \in \{\langle\!\langle x \rangle\!\rangle, [\![x]\!]\}$.

A *binding prefix* over a set of variables $\mathrm{V} \subseteq \mathrm{Vr}$ 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 $\flat$, i.e., there is exactly one index $i \in [0, |\mathrm{Ag}|[$ for which $(\flat)_i \in \{(a, x) : x \in \mathrm{V}\}$.

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

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

**Definition 2.1.1 ($\mathrm{SL}[\mathrm{NG}]$, $\mathrm{SL}[\mathrm{BG}]$, and $\mathrm{SL}[1\mathrm{G}]$ Syntax)** *$\mathrm{SL}[\mathrm{NG}]$ formulas are built inductively from the sets of atomic propositions* AP*, quantification prefixes* $\mathrm{Qnt}(\mathrm{V})$ *for any* $\mathrm{V} \subseteq \mathrm{Vr}$*, and binding prefixes* $\mathrm{Bnd}(\mathrm{Vr})$*, by using the following grammar, with* $p \in \mathrm{AP}$*,* $\wp \in \cup_{\mathrm{V} \subseteq \mathrm{Vr}} \mathrm{Qnt}(\mathrm{V})$*, and* $\flat \in \mathrm{Bnd}(\mathrm{Vr})$*:*

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

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

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

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

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

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

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

We want to remark that the nesting of goals in SL formulas is much more expressive than the nesting of quantification in ATL$^\star$. Indeed, in ATL$^\star$, we can make only a nesting of quantifications that are in some sense complete w.r.t. players, which implies, at every nesting, a total replacement of the strategy profile for all players. This is due to the fact that the quantifier operator over a set of agents A in ATL$^\star$ includes implicitly the quantification of strategies for agents not in A, i.e., each quantifier ranges over the whole set of agents, and the binding is immediately applied to the agents. Instead, in the case of SL, the quantification is referred to a single strategy and the binding with an agent can be postponed. This power of nesting allows to embed very expressive logics such as QPTL, as it is shown in Section 1.2.1

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

To start to practice with the above fragments, consider again the sentence $\varphi$ of Example 1.1.11 on page 12. It is easy to see that it actually belongs to SL[BG, 2-AG, 3-VAR, 2-ALT], and so, to SL[NG], but not to SL[1G], since it is of the form $\wp(\flat_1 \mathtt{Xp} \wedge \flat_2 \mathtt{Xq})$, where the quantification prefix is $\wp = \langle\langle \mathtt{x} \rangle\rangle [\![ \mathtt{y} ]\!] \langle\langle \mathtt{z} \rangle\rangle$ and the binding prefixes of the two goals are $\flat_1 = (\alpha, \mathtt{x})(\beta, \mathtt{y})$ and $\flat_2 = (\alpha, \mathtt{y})(\beta, \mathtt{z})$.

Along the paper, sometimes we assert that a given formula $\varphi$ belongs to an SL syntactic fragment also if its syntax does not precisely correspond to what is described by the relative grammar. We do this in order to make easier the reading and interpretation of the formula $\varphi$ itself and only in the case that it is simple to translate it into an equivalent formula that effectively belongs to the intended logic, by means of a simple generalization of classic rules used to put a formula of first order logic in the prenex normal form. For example, consider the sentence $\varphi_{NE}$ of Example 1.1.6 on page 6 representing the existence of a Nash equilibrium. This formula is considered to belong to SL[BG, $n$-AG, $2n$-VAR, FVS, 1-ALT], since it can be easily translated in the form $\phi_{NE} = \wp \bigwedge_{i=1}^{n} \flat_i \psi_i \rightarrow \flat \psi_i$, where $\wp = \langle\langle \mathtt{x}_1 \rangle\rangle \cdots \langle\langle \mathtt{x}_n \rangle\rangle [\![ \mathtt{y}_1 ]\!] \cdots [\![ \mathtt{y}_n ]\!]$, $\flat = (\alpha_1, \mathtt{x}_1) \cdots (\alpha_n, \mathtt{x}_n)$, $\flat_i = (\alpha_1, \mathtt{x}_1) \cdots (\alpha_{i-1}, \mathtt{x}_{i-1})(\alpha_i, \mathtt{y}_i)(\alpha_{i+1}, \mathtt{x}_{i+1}) \cdots (\alpha_n, \mathtt{x}_n)$, and $\mathsf{free}(\psi_i) = \mathrm{Ag}$. As another example, consider the sentence $\varphi_{SP}$ of Example 1.1.7 on page 7 representing the existence of a stability profile. Also this formula is considered to belong to SL[BG, $n$-AG, $2n$-VAR, FVS, 1-ALT], since it is equivalent to $\phi_{SP} = \wp \bigwedge_{i,j=1, i \neq j}^{n} \flat \psi_j \rightarrow ((\flat \psi_i \leftrightarrow \flat_i \psi_i) \rightarrow \flat_i \psi_j)$. Furthermore, it is immediate to see that, Example 1.1.15 on page 13, the formulas $\varphi_1$ and $\varphi_2$ are in SL[1G, 4-AG, 4-VAR, FVS, 1-ALT], while $\varphi_3$ is in SL[1G, 4-AG, 4-VAR, FVS, 2-ALT] Note that both $\phi_{NE}$ and $\phi_{SP}$ are principal sentences.

Now, it is interesting to observe that CTL$^\star$ and ATL$^\star$ are exactly equivalent to SL[1G, FVS, 0-ALT] and SL[1G, FVS, 1-ALT], respectively. Moreover, GL [AHK02] is the very simple fragment of SL[BG, FVS, 1-ALT] that forces all goals in a formula to have a common part containing all variables quantified before the unique possible alternation of

the quantification prefix. Finally, we have that CHP-SL is the SL[BG, 2-AG, FVS] fragment interpreted over turn-based CGS.

From now on, we refer to CTL$^\star$, ATL$^\star$, and CHP-SL formulas by denoting them with their translation in SL.

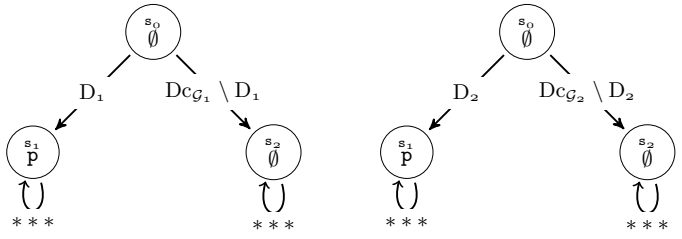
**Remark 2.1.1 (SL[NG] Model-Checking Hardness)** *It is well-known that the non-elementary hardness result for the satisfiability problem of* QPTL *[SVW87] already holds for formulas in prenex normal form. Now, it is not hard to see that the transformation described in Lemma 1.2.1 on page 17 of* QPTL *reduction puts* QPTL[$k$-ALT] *sentences $\varphi$ in prenex normal form into* SL[NG, 1-AG, $k$-ALT] *variable-closed formulas $\overline{\varphi} = \wp\psi$. Moreover, the derived* SL[1-AG, $k$-ALT] *sentence $\langle\langle \mathtt{x} \rangle\rangle(\alpha, \mathtt{x})\wp\psi$ used in Theorem 1.2.1 on page 18 of* SL *model-checking hardness is equivalent to the* SL[NG, 1-AG, $k$-ALT] *principal sentence $\langle\langle \mathtt{x} \rangle\rangle\wp(\alpha, \mathtt{x})\psi$, since $\mathtt{x}$ is not used in the quantification prefix $\wp$. Thus, the hardness result for the model-checking problem holds for* SL[NG, 1-AG, $k$-ALT] *as well. However, it is important to observe that, unfortunately, it is not know if such an hardness result holds for* SL[BG] *and, in particular, for* CHP-SL.

**Remark 2.1.2 (SL[BG] Satisfiability)** *In Section 1.3.2 on page 21, we prove the undecidability of the satisfiability problem for* SL. *Now, it is not hard to see that the formula $\varphi^{dom}$ used to reduce the domino problem in Theorem 1.3.2 on page 24 actually lies in the* SL[BG] *fragment. Hence, the satisfiability for this logic is undecidable too. On the other hand, later in the thesis, we prove that the same problem for* SL[1G] *is 2*EXPTIME-COMPLETE*, thus not harder than the one for* ATL$^\star$.

Differently from SL[1G], in fragments like SL[BG] we are able to associate different strategies with the same agent. This feature allows to represent game properties such as Nash Equilibria, Equilibrium Profiles, and the like. This syntactic flexibility, however, is not the feature that forces these fragments to have non-elementary complexities for the model-checking problem. Indeed, as it has been recently pointed out in [MMS13], the same problem for some of the SL fragments preserving this feature is 2EXPTIME-COMPLETE. Thus, the non-elementary hardness in the model-checking problem for SL[NG] is due to the ability to nest the binding of strategies by means of temporal operators. A deeper study of these aspects is out of the scope of this paper.

At this point, we prove that ATL$^\star$ is strictly less expressive than SL[1G] and, consequently, than SL[BG] and SL[NG]. To do this, we show the existence of two structures that result to be equivalent only *w.r.t.* sentences having alternation number bounded by 1. It can be interesting to note that, we use an ad-hoc technique based on a brute-force check to verify that all ATL$^\star$ formulas cannot distinguish between the two structures. A possible future line of research is to study variants of the Ehrenfeucht-Fraïssé game [EF95, Hod93] for SL, which would allow to determine in a more direct way whether two structures are equivalent or not *w.r.t.* a particular SL fragment.

**Theorem 2.1.1 (SL[1G] vs ATL$^\star$ Expressiveness)** *There exists an* SL[1G] *sentence having no* ATL$^\star$ *equivalent sentence.*




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

Figure 2.1: Alternation-2 non-equivalent CGS s.

*Proof.* Consider the two CGS s $\mathcal{G}_1 \triangleq \langle\{\mathtt{p}\}, \{\alpha, \beta, \gamma\}, \{0, 1\}, \{\mathtt{s}_0, \mathtt{s}_1, \mathtt{s}_2\}, \mathsf{L}, \mathtt{tr}_1, \mathtt{s}_0\rangle$ and $\mathcal{G}_2 \triangleq \langle\{\mathtt{p}\}, \{\alpha, \beta, \gamma\}, \{0, 1, 2\}, \{\mathtt{s}_0, \mathtt{s}_1, \mathtt{s}_2\}, \mathsf{L}, \mathtt{tr}_2, \mathtt{s}_0\rangle$ depicted in Figure 2.1, where $\mathsf{L}(\mathtt{s}_0) = \mathsf{L}(\mathtt{s}_2) \triangleq \emptyset$, $\mathsf{L}(\mathtt{s}_1) \triangleq \{\mathtt{p}\}$, $\mathrm{D}_1 \triangleq \{00*, 11*\}$, and $\mathrm{D}_2 \triangleq \{00*, 11*, 12*, 200, 202, 211\}$. Moreover, consider the SL[1G] sentence $\varphi^* \triangleq \wp^*\flat^*\mathtt{Xp}$, where $\wp^* \triangleq [\![\mathtt{x}]\!]\langle\!\langle\mathtt{y}\rangle\!\rangle[\![z]\!]$ and $\flat^* \triangleq (\alpha, \mathtt{x})(\beta, \mathtt{y})(\gamma, \mathtt{z})$. Then, it is easy to see that $\mathcal{G}_1 \models \varphi^*$ but $\mathcal{G}_2 \not\models \varphi^*$. Indeed, $\mathcal{G}_1, \chi_1, \mathtt{s}_0 \models \flat^*\mathtt{Xp}$, for all $\chi_1 \in \mathrm{Asg}_{\mathcal{G}_1}(\{\mathtt{x}, \mathtt{y}, \mathtt{z}\}, \mathtt{s}_0)$ such that $\chi_1(\mathtt{y})(\mathtt{s}_0) = \chi_1(\mathtt{x})(\mathtt{s}_0)$, and $\mathcal{G}_2, \chi_2, \mathtt{s}_0 \models \flat^*\mathtt{X}\neg\mathtt{p}$, for all $\chi_2 \in \mathrm{Asg}_{\mathcal{G}_2}(\{\mathtt{x}, \mathtt{y}, \mathtt{z}\}, \mathtt{s}_0)$ such that $\chi_2(\mathtt{x})(\mathtt{s}_0) = 2$ and $\chi_2(\mathtt{z})(\mathtt{s}_0) = (\chi_2(\mathtt{y})(\mathtt{s}_0) + 1) \bmod 3$.

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

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

## 2.2 Behavioral property

### 2.2.1 Skolem Dependence Functions

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

First, we introduce some notation regarding quantification prefixes. Let $\wp \in \mathrm{Qnt}(\mathrm{V})$ be a quantification prefix over a set $\mathrm{V}(\wp) \triangleq \mathrm{V} \subseteq \mathrm{Vr}$ of variables. By $\langle\!\langle\wp\rangle\!\rangle \triangleq \{x \in \mathrm{V}(\wp) : \exists i \in [0, |\wp|[.\ (\wp)_i = \langle\!\langle x\rangle\!\rangle\}$ and $[\![\wp]\!] \triangleq \mathrm{V}(\wp) \setminus \langle\!\langle\wp\rangle\!\rangle$ we denote the sets of *existential*

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

Finally, we define the notion of *valuation* of variables over a generic set D, called *domain*, i.e., a partial function $\mathsf{v} : \mathrm{Vr} \rightharpoonup \mathrm{D}$ 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 $\mathrm{V} \subseteq \mathrm{Vr}$.

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

**Definition 2.2.1 (Skolem Dependence Function)** *Let $\wp \in \mathrm{Qnt}(\mathrm{V})$ be a quantification prefix over a set $\mathrm{V} \subseteq \mathrm{Vr}$ of variables, and $\mathrm{D}$ a set. Then, a* Skolem dependence function *for $\wp$ over $\mathrm{D}$ is a function $\theta : \mathrm{Val}_\mathrm{D}([\![\wp]\!]) \to \mathrm{Val}_\mathrm{D}(\mathrm{V})$ satisfying the following properties:*

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

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

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

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

for all $\mathtt{v} \in \mathrm{Val_D}([\![\wp]\!])$, and $\overline{\theta_i}(\overline{\mathtt{v}})(\mathtt{x}) = 0$ with $i \in [0,3]$, $\overline{\theta_i}(\overline{\mathtt{v}})(\mathtt{x}) = 1$ with $i \in [4,7]$, $\overline{\theta_0}(\overline{\mathtt{v}})(\mathtt{z}) = \overline{\theta_4}(\overline{\mathtt{v}})(\mathtt{z}) = 0$, $\overline{\theta_1}(\overline{\mathtt{v}})(\mathtt{z}) = \overline{\theta_5}(\overline{\mathtt{v}})(\mathtt{z}) = \overline{\mathtt{v}}(\mathtt{y})$, $\overline{\theta_2}(\overline{\mathtt{v}})(\mathtt{z}) = \overline{\theta_6}(\overline{\mathtt{v}})(\mathtt{z}) = 1 - \overline{\mathtt{v}}(\mathtt{y})$, and $\overline{\theta_3}(\overline{\mathtt{v}})(\mathtt{z}) = \overline{\theta_7}(\overline{\mathtt{v}})(\mathtt{z}) = 1$, for all $\overline{\mathtt{v}} \in \mathrm{Val_D}([\![\overline{\wp}]\!])$.

We now prove the following fundamental theorem that describes how to eliminate the strategy quantifications of an SL formula via a choice of a suitable Skolem dependence function over strategies. This procedure can be seen as the equivalent of *Skolemization* in first order logic (see [Hod93], for more details).

**Theorem 2.2.1 (SL Strategy Quantification)** *Let $\mathcal{G}$ be a* CGS *and $\varphi = \wp\psi$ an* SL *formula, being $\wp \in \mathrm{Qnt(V)}$ a quantification prefix over a set $\mathrm{V} \subseteq \mathsf{free}(\psi) \cap \mathrm{Vr}$ of variables. Then, for all assignments $\chi \in \mathrm{Asg}(\mathsf{free}(\varphi), s_\mathrm{o})$, the following holds: $\mathcal{G}, \chi, s_\mathrm{o} \models \varphi$ iff there exists a Skolem dependence function $\theta \in \mathrm{SDF}_{\mathrm{Str}(s_\mathrm{o})}(\wp)$ such that $\mathcal{G}, \chi \Cup \theta(\chi'), s_\mathrm{o} \models \psi$, for all $\chi' \in \mathrm{Asg}([\![\wp]\!], s_\mathrm{o})$.* [2]

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

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

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

- $\mathtt{Qn} = \langle\!\langle x \rangle\!\rangle$.

  By Item 3a of Definition 1.1.6 on page 10 of SL semantics, there is a strategy $\mathsf{f} \in \mathrm{Str}(s_\mathrm{o})$ such that $\mathcal{G}, \chi^{x \mapsto \mathsf{f}}, s_\mathrm{o} \models \wp'\psi$. Note that $[\![\wp]\!] = [\![\wp']\!]$. By the inductive hypothesis, we have that there exists a Skolem dependence function $\theta \in \mathrm{SDF}_{\mathrm{Str}(s_\mathrm{o})}(\wp')$ such that $\mathcal{G}, \chi^{x \mapsto \mathsf{f}} \Cup \theta(\chi'), s_\mathrm{o} \models \psi$, for all $\chi' \in \mathrm{Asg}([\![\wp']\!], s_\mathrm{o})$. Now, consider the function $\theta' : \mathrm{Asg}([\![\wp]\!], s_\mathrm{o}) \to \mathrm{Asg(V}, s_\mathrm{o})$ defined by $\theta'(\chi') \triangleq \theta(\chi')[x \mapsto \mathsf{f}]$, for all $\chi' \in \mathrm{Asg}([\![\wp]\!], s_\mathrm{o})$. It is easy to check that $\theta'$ is a Skolem dependence function for $\wp$ over $\mathrm{Str}(s_\mathrm{o})$, i.e., $\theta' \in \mathrm{SDF}_{\mathrm{Str}(s_\mathrm{o})}(\wp)$. Moreover, $\chi^{x \mapsto \mathsf{f}} \Cup \theta(\chi') = \chi \Cup \theta(\chi')[x \mapsto \mathsf{f}] = \chi \Cup \theta'(\chi')$, for $\chi' \in \mathrm{Asg}([\![\wp]\!], s_\mathrm{o})$. Hence, $\mathcal{G}, \chi \Cup \theta'(\chi'), s_\mathrm{o} \models \psi$, for all $\chi' \in \mathrm{Asg}([\![\wp]\!], s_\mathrm{o})$.

- $\mathtt{Qn} = [\![x]\!]$.

  By Item 3b of Definition 1.1.6 on page 10, we have that, for all strategies $\mathsf{f} \in \mathrm{Str}(s_\mathrm{o})$, it holds that $\mathcal{G}, \chi^{x \mapsto \mathsf{f}}, s_\mathrm{o} \models \wp'\psi$. Note that $[\![\wp]\!] = [\![\wp']\!] \cup \{x\}$. By the inductive hypothesis, we derive that, for each $\mathsf{f} \in \mathrm{Str}(s_\mathrm{o})$, there exists a Skolem dependence function $\theta^{\mathsf{f}} \in \mathrm{SDF}_{\mathrm{Str}(s_\mathrm{o})}(\wp')$ such that $\mathcal{G}, \chi^{x \mapsto \mathsf{f}} \Cup \theta^{\mathsf{f}}(\chi'), s_\mathrm{o} \models \psi$, for all $\chi' \in \mathrm{Asg}([\![\wp']\!], s_\mathrm{o})$. Now, consider the function $\theta' : \mathrm{Asg}([\![\wp]\!], s_\mathrm{o}) \to \mathrm{Asg(V}, s_\mathrm{o})$ defined by $\theta'(\chi') \triangleq \theta^{\chi'(x)}(\chi_{|'[\![\wp']\!]})[x \mapsto \chi'(x)]$, for all $\chi' \in \mathrm{Asg}([\![\wp]\!], s_\mathrm{o})$. It is evident that $\theta'$ is a Skolem dependence function for $\wp$ over $\mathrm{Str}(s_\mathrm{o})$, i.e., $\theta' \in \mathrm{SDF}_{\mathrm{Str}(s_\mathrm{o})}(\wp)$. Moreover, $\chi^{x \mapsto \mathsf{f}} \Cup \theta^{\mathsf{f}}(\chi') = \chi \Cup \theta^{\mathsf{f}}(\chi')[x \mapsto \mathsf{f}] = \chi \Cup \theta'(\chi'[x \mapsto \mathsf{f}])$, for $\mathsf{f} \in \mathrm{Str}(s_\mathrm{o})$ and $\chi' \in \mathrm{Asg}([\![\wp']\!], s_\mathrm{o})$. Hence, $\mathcal{G}, \chi \Cup \theta'(\chi'), s_\mathrm{o} \models \psi$, for all $\chi' \in \mathrm{Asg}([\![\wp]\!], s_\mathrm{o})$.

[2] By $\mathrm{g_1} \Cup \mathrm{g_2} : (\mathrm{X_1} \cup \mathrm{X_2}) \to (\mathrm{Y_1} \cup \mathrm{Y_2})$ we denote the operation of *union* of two functions $\mathrm{g_1} : \mathrm{X_1} \to \mathrm{Y_1}$ and $\mathrm{g_2} : \mathrm{X_2} \to \mathrm{Y_2}$ defined on disjoint domains, i.e., $\mathrm{X_1} \cap \mathrm{X_2} = \emptyset$.

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

- $\mathtt{Qn} = \langle\!\langle x \rangle\!\rangle$.

  There is a strategy $\mathsf{f} \in \mathrm{Str}(s_\mathrm{o})$ such that $\mathsf{f} = \theta(\chi')(x)$, for all $\chi' \in \mathrm{Asg}([\![\wp]\!], s_\mathrm{o})$. Note that $[\![\wp]\!] = [\![\wp']\!]$. Consider the function $\theta' : \mathrm{Asg}([\![\wp']\!], s_\mathrm{o}) \to \mathrm{Asg}(\mathrm{V} \setminus \{x\}, s_\mathrm{o})$ defined by $\theta'(\chi') \triangleq \theta(\chi')_{\restriction(\mathrm{V}\setminus\{x\})}$, for all $\chi' \in \mathrm{Asg}([\![\wp']\!], s_\mathrm{o})$. It is easy to check that $\theta'$ is a Skolem dependence function for $\wp'$ over $\mathrm{Str}(s_\mathrm{o})$, i.e., $\theta' \in \mathrm{SDF}_{\mathrm{Str}(s_\mathrm{o})}(\wp')$. Moreover, $\chi \uplus \theta(\chi') = \chi \uplus \theta'(\chi')[x \mapsto \mathsf{f}] = \chi^{x \mapsto \mathsf{f}} \uplus \theta'(\chi')$, for $\chi' \in \mathrm{Asg}([\![\wp']\!], s_\mathrm{o})$. Then, it is evident that $\mathcal{G}, \chi^{x \to \mathsf{f}} \uplus \theta'(\chi'), s_\mathrm{o} \models \psi$, for all $\chi' \in \mathrm{Asg}([\![\wp']\!], s_\mathrm{o})$. By the inductive hypothesis, we derive that $\mathcal{G}, \chi^{x \mapsto \mathsf{f}}, s_\mathrm{o} \models \wp'\psi$, which means that $\mathcal{G}, \chi, s_\mathrm{o} \models \varphi$, by Item 3a of Definition 1.1.6 on page 10 of SL semantics.

- $\mathtt{Qn} = [\![x]\!]$.

  First note that $[\![\wp]\!] = [\![\wp']\!] \cup \{x\}$. Also, consider the functions $\theta^{\mathsf{f}|'} : \mathrm{Asg}([\![\wp']\!], s_\mathrm{o}) \to \mathrm{Asg}(\mathrm{V} \setminus \{x\}, s_\mathrm{o})$ defined by $\theta^{\mathsf{f}|'}(\chi') \triangleq \theta(\chi'[x \mapsto \mathsf{f}])_{\restriction(\mathrm{V}\setminus\{x\})}$, for each $\mathsf{f} \in \mathrm{Str}(s_\mathrm{o})$ and $\chi' \in \mathrm{Asg}([\![\wp']\!], s_\mathrm{o})$. It is easy to see that every $\theta^{\mathsf{f}|'}$ is a Skolem dependence function for $\wp'$ over $\mathrm{Str}(s_\mathrm{o})$, i.e., $\theta^{\mathsf{f}|'} \in \mathrm{SDF}_{\mathrm{Str}(s_\mathrm{o})}(\wp')$. Moreover, $\chi \uplus \theta(\chi') = \chi \uplus \theta^{\chi'(x)|'}(\chi'_{\restriction[\![\wp']\!]})[x \mapsto \chi'(x)] = \chi^{x \mapsto \chi'(x)} \uplus \theta^{\chi'(x)|'}(\chi'_{\restriction[\![\wp']\!]})$, for $\chi' \in \mathrm{Asg}([\![\wp]\!], s_\mathrm{o})$. Then, it is evident that $\mathcal{G}, \chi^{x \to \mathsf{f}} \uplus \theta^{\mathsf{f}|'}(\chi'), s_\mathrm{o} \models \psi$, for all $\mathsf{f} \in \mathrm{Str}(s_\mathrm{o})$ and $\chi' \in \mathrm{Asg}([\![\wp']\!], s_\mathrm{o})$. By the inductive hypothesis, we derive that $\mathcal{G}, \chi^{x \mapsto \mathsf{f}}, s_\mathrm{o} \models \wp'\psi$, for all $\mathsf{f} \in \mathrm{Str}(s_\mathrm{o})$, which means that $\mathcal{G}, \chi, s_\mathrm{o} \models \varphi$, by Item 3b of Definition 1.1.6 on page 10.

Thus, the thesis of the theorem holds. □

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

**Corollary 2.2.1 (SL Strategy Quantification)** *Let $\mathcal{G}$ be a* CGS *and $\varphi = \wp\psi$ an* SL *sentence, where $\psi$ is agent-closed and $\wp \in \mathrm{Qnt}(\mathsf{free}(\psi))$. Then, $\mathcal{G} \models \varphi$ iff there exists a Skolem dependence function $\theta \in \mathrm{SDF}_{\mathrm{Str}(s_\mathrm{o})}(\wp)$ such that $\mathcal{G}, \theta(\chi), s_\mathrm{o} \models \psi$, for all $\chi \in \mathrm{Asg}([\![\wp]\!], s_\mathrm{o})$.*

### 2.2.2 Behavioral quantifications

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

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

**Definition 2.2.2 (Adjoint Functions)** *Let* $\mathrm{D}$, $\mathrm{T}$, $\mathrm{U}$, *and* $\mathrm{V}$ *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* $\mathsf{m}$ *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* $t \in \mathrm{T}$ [3]

Intuitively, $\widetilde{\mathsf{m}}$ is the adjoint of $\mathsf{m}$ if the dependence from the set $\mathrm{T}$ in both domain and codomain of the latter can be extracted and put as a common factor of the functor given by the former. This means also that, for every pair of functions $\mathsf{g}_1, \mathsf{g}_2 \in (\mathrm{T} \to \mathrm{D})^{\mathrm{U}}$ such that $\widehat{\mathsf{g}_1}(t) = \widehat{\mathsf{g}_2}(t)$ for some $t \in \mathrm{T}$, it holds that $\mathsf{m}(\mathsf{g}_1)(x)(t) = \mathsf{m}(\mathsf{g}_2)(x)(t)$, for all $x \in \mathrm{V}$. It is immediate to observe that if a function has an adjoint then this adjoint is unique. At the same way, if one has an adjoint function then it is possible to determine the original function without any ambiguity. Thus, it is established a one-to-one correspondence between functions admitting an adjoint and the adjoint itself.

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

**Lemma 2.2.1 (Adjoint Skolem Dependence Functions)** *Let* $\wp \in \mathrm{Qnt}(\mathrm{V})$ *be a quantification prefix over a set* $\mathrm{V} \subseteq \mathrm{Vr}$ *of variables,* $\mathrm{D}$ *and* $\mathrm{T}$ *two sets, and* $\theta : \mathrm{Val}_{\mathrm{T}\to\mathrm{D}}([\![\wp]\!]) \to \mathrm{Val}_{\mathrm{T}\to\mathrm{D}}(\mathrm{V})$ *and* $\widetilde{\theta} : \mathrm{T} \to (\mathrm{Val}_{\mathrm{D}}([\![\wp]\!]) \to \mathrm{Val}_{\mathrm{D}}(\mathrm{V}))$ *two functions such that* $\widetilde{\theta}$ *is the adjoint of* $\theta$*. Then,* $\theta \in \mathrm{SDF}_{\mathrm{T}\to\mathrm{D}}(\wp)$ *iff, for all* $t \in \mathrm{T}$*, it holds that* $\widetilde{\theta}(t) \in \mathrm{SDF}_{\mathrm{D}}(\wp)$.

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

**Definition 2.2.3 (Behavioral Skolem Dependence Functions)** *Let* $\wp \in \mathrm{Qnt}(\mathrm{V})$ *be a quantification prefix over a set* $\mathrm{V} \subseteq \mathrm{Vr}$ *of variables,* $\mathrm{D}$ *and* $\mathrm{T}$ *two sets, and* $\theta \in \mathrm{SDF}_{\mathrm{T}\to\mathrm{D}}(\wp)$ *a Skolem dependence function for* $\wp$ *over* $\mathrm{T} \to \mathrm{D}$*. Then,* $\theta$ *is* behavioral *if it admits an adjoint function.* $\mathrm{BSDF}_{\mathrm{T}\to\mathrm{D}}(\wp)$ *denotes the set of all behavioral Skolem dependence functions for* $\wp$ *over* $\mathrm{T} \to \mathrm{D}$.

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

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

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

**Definition 2.2.4 (**SL[NG] **Behavioral Semantics)** *Let $\mathcal{G}$ be a* CGS*, $s \in \mathrm{St}$ one of its states, and $\wp\psi$ an* SL[NG] *formula, where $\psi$ is agent-closed and $\wp \in \mathrm{Qnt}(\mathsf{free}(\psi))$. Then $\mathcal{G}, \varnothing, s \models_{\mathrm{B}} \wp\psi$ if there is a behavioral Skolem dependence function $\theta \in \mathrm{BSDF}_{\mathrm{Str}(s)}(\wp)$ for $\wp$ over $\mathrm{Str}(s)$ such that $\mathcal{G}, \theta(\chi), s \models_{\mathrm{B}} \psi$, for all $\chi \in \mathrm{Asg}([\![\wp]\!], s)$.*

It is immediate to see a strong similarity between the statement of Corollary 2.2.1 on page 33 of SL strategy quantification and the previous definition. The only crucial difference resides in the choice of the kind of Skolem dependence function. Moreover, observe that, differently from the classic semantics, the quantifications in the prefix are not treated individually but as an atomic block. This is due to the necessity of having a strict correlation between the point-wise structure of the quantified strategies.

**Remark 2.2.1 (**SL **Behavioral Semantics)** *It can be interesting to know that we do not define a behavioral semantics for the whole* SL*, since we are not able, at the moment, to easily use the concept of behavioral Skolem dependence function, when the quantifications are not necessarily grouped in prefixes, i.e., when the formula is not in prenex normal form. In fact, this may represent a challenging problem, whose solution is left to future works.*

Due to the new semantics of SL[NG], we have to redefine the related concepts of model and satisfiability, in order to differentiate between the classic relation $\models$ and the behavioral one $\models_{\mathrm{B}}$. Indeed, as we show later, there are sentences that are satisfiable but not behavioral satisfiable. We say that a CGS $\mathcal{G}$ is a *behavioral model* of an SL[NG] sentence $\varphi$, in symbols $\mathcal{G} \models_{\mathrm{B}} \varphi$, if $\mathcal{G}, \varnothing, s_{\mathrm{o}} \models_{\mathrm{B}} \varphi$. In general, we also say that $\mathcal{G}$ is a *behavioral model* for $\varphi$ on $s \in \mathrm{St}$, in symbols $\mathcal{G}, s \models_{\mathrm{B}} \varphi$, if $\mathcal{G}, \varnothing, s \models_{\mathrm{B}} \varphi$. An SL[NG] sentence $\varphi$ is *behavioral satisfiable* if there is an behavioral model for it.

We have to modify the concepts of implication and equivalence, as well. Indeed, also in this case we can have pairs of equivalent formulas that are not behavioral equivalent, and vice versa. Thus, we have to be careful when we use natural transformation between

formulas, since it can be the case that they preserve the meaning only under the classic semantics. An example of this problem can arise when one want to put a formula in *pnf*. Given two SL[NG] formulas $\varphi_1$ and $\varphi_2$ with $\mathsf{free}(\varphi_1) = \mathsf{free}(\varphi_2)$, we say that $\varphi_1$ *behaviorally implies* $\varphi_2$, in symbols $\varphi_1 \Rightarrow_{\mathrm{B}} \varphi_2$, if, for all CGS s $\mathcal{G}$, states $s \in \mathrm{St}$, and $\mathsf{free}(\varphi_1)$-defined $s$-total assignments $\chi \in \mathrm{Asg}(\mathsf{free}(\varphi_1), s)$, it holds that if $\mathcal{G}, \chi, s \models_{\mathrm{B}} \varphi_1$ then $\mathcal{G}, \chi, s \models_{\mathrm{B}} \varphi_2$. Accordingly, we say that $\varphi_1$ is *behaviorally equivalent* to $\varphi_2$, in symbols $\varphi_1 \equiv_{\mathrm{B}} \varphi_2$, if both $\varphi_1 \Rightarrow_{\mathrm{B}} \varphi_2$ and $\varphi_2 \Rightarrow_{\mathrm{B}} \varphi_1$ hold.

### 2.2.3 Behavioral and non-behavioral semantics

Finally, we show that the introduced concept of behavioral satisfiability is relevant to the context of our logic, as its applicability represents a demarcation line between "easy" and "hard" fragments of SL. Moreover, we believe that it is because of this fundamental property that several well-known temporal logics are so robustly decidable [Var96].

**Remark 2.2.2 (**SL[NG, 0-ALT] **Behavioral)** *It is interesting to observe that, for every* CGS $\mathcal{G}$ *and* SL[NG, 0-ALT] *sentence* $\varphi$*, it holds that* $\mathcal{G} \models \varphi$ *iff* $\mathcal{G} \models_{\mathrm{B}} \varphi$*. This is an immediate consequence of the fact that all quantification prefixes* $\wp$ *used in* $\varphi$ *belong to* $\mathrm{Qnt}^{\exists^*\forall^*}(\mathrm{V})$*, for a given set* $\mathrm{V} \subseteq \mathrm{Vr}$ *of variables. Thus, as already mentioned, the related Skolem dependence functions on strategies* $\theta \in \mathrm{SDF}_{\mathrm{Str}(s_{\mathrm{o}})}(\wp)$ *are necessarily behavioral.*

By Corollary 2.2.1 on page 33 of SL strategy quantification, it is easy to see that the following coherence property about the behavioral of the SL[NG] satisfiability holds. Intuitively, it asserts that every behaviorally satisfiable sentence in *pnf* is satisfiable too.

**Theorem 2.2.2 (**SL[NG] **Behavioral Coherence)** *Let* $\mathcal{G}$ *be a* CGS*,* $s \in \mathrm{St}$ *one of its states,* $\varphi$ *an* SL[NG] *formula in* pnf*, and* $\chi \in \mathrm{Asg}(s)$ *an* $s$*-total assignment with* $\mathsf{free}(\varphi) \subseteq \mathsf{dom}(\chi)$*. Then, it holds that* $\mathcal{G}, \chi, s \models_{\mathrm{B}} \varphi$ *implies* $\mathcal{G}, \chi, s \models \varphi$*.*

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

Suppose that $\mathcal{G}, \varnothing, s \models_{\mathrm{B}} \wp\psi$. Then, by Definition 2.2.4 on the previous page of SL[NG] behavioral semantics, there is a behavioral Skolem dependence function $\theta \in \mathrm{BSDF}_{\mathrm{Str}(s)}(\wp)$ such that, for all assignments $\chi \in \mathrm{Asg}([\![\wp]\!], s)$, it holds that $\mathcal{G}, \theta(\chi), s \models_{\mathrm{B}} \psi$. Now, by the inductive hypothesis, there is a Skolem dependence function $\theta \in \mathrm{SDF}_{\mathrm{Str}(s)}(\wp)$ such that, for all assignments $\chi \in \mathrm{Asg}([\![\wp]\!], s)$, it holds that $\mathcal{G}, \theta(\chi), s \models \psi$. Hence, by Corollary 2.2.1 on page 33 of SL strategy quantification, we have that $\mathcal{G}, \varnothing, s \models \wp\psi$. □

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

**Theorem 2.2.3 (**SL[BG] **Non-Behavioral)** *There exists a satisfiable* SL[BG, 1-AG, 2-VAR, 1-ALT] *sentence in* pnf *that is not behaviorally satisfiable.*

*Proof.* Consider the SL[BG, 1-AG, 2-VAR, 1-ALT] sentence $\varphi \triangleq \varphi_1 \wedge \varphi_2$ in *pnf* where $\varphi_1 \triangleq \wp(\psi_1 \wedge \psi_2)$, with $\wp \triangleq [\![\mathtt{x}]\!]\langle\!\langle \mathtt{y} \rangle\!\rangle$, $\psi_1 \triangleq (\alpha, \mathtt{x})\mathtt{Xp} \leftrightarrow (\alpha, \mathtt{y})\mathtt{X}\neg\mathtt{p}$, and $\psi_2 \triangleq (\alpha, \mathtt{x})\mathtt{XXp} \leftrightarrow (\alpha, \mathtt{y})\mathtt{XXp}$, and $\varphi_2 \triangleq [\![\mathtt{x}]\!](\alpha, \mathtt{x})\mathtt{X}((\langle\!\langle \mathtt{x} \rangle\!\rangle(\alpha, \mathtt{x})\mathtt{Xp}) \wedge (\langle\!\langle \mathtt{x} \rangle\!\rangle(\alpha, \mathtt{x})\mathtt{X}\neg\mathtt{p}))$. Moreover, note that the SL[1G, 1-AG, 1-VAR, 0-ALT] sentence $\varphi_2$ is equivalent to the CTL formula $\mathtt{AX}((\mathtt{EXp}) \wedge (\mathtt{EX}\neg\mathtt{p}))$. Then, it is easy to see that the turn-based CGS $\mathcal{G}_{Rdc}$ of Figure 1.8 on page 17 satisfies $\varphi$. Indeed, $\mathcal{G}_{Rdc}, \theta(\chi), \mathtt{s}_0 \models \psi_1 \wedge \psi_2$, for all assignments $\chi \in \mathrm{Asg}(\{\mathtt{x}\}, \mathtt{s}_0)$, where the non-behavioral Skolem dependence function $\theta \in \mathrm{SDF}_{\mathrm{Str}(\mathtt{s}_0)}(\wp)$ is such that $\theta(\chi)(\mathtt{y})(\mathtt{s}_0) = \neg\chi(\mathtt{x})(\mathtt{s}_0)$ and $\theta(\chi)(\mathtt{y})(\mathtt{s}_0 \cdot s_i) = \chi(\mathtt{x})(\mathtt{s}_0 \cdot s_{1-i})$, for all $i \in \{0, 1\}$.

Now, let $\mathcal{G}$ be a generic CGS. If $\mathcal{G} \not\models \varphi$, by Theorem 2.2.2 on the facing page of SL[NG] behavioral coherence, it holds that $\mathcal{G} \not\models_{\mathrm{B}} \varphi$. Otherwise, we have that $\mathcal{G} \models \varphi$ and, in particular, $\mathcal{G} \models \varphi_1$, which means that $\mathcal{G} \models \wp(\psi_1 \wedge \psi_2)$. At this point, to prove that $\mathcal{G} \not\models_{\mathrm{B}} \varphi$, we show that, for all behavioral Skolem dependence functions $\theta \in \mathrm{BSDF}_{\mathrm{Str}(s_0)}(\wp)$, there exists an assignment $\chi \in \mathrm{Asg}(\{\mathtt{x}\}, s_0)$ such that $\mathcal{G}, \theta(\chi), s_0 \not\models_{\mathrm{B}} \psi_1 \wedge \psi_2$. To do this, let us fix a behavioral Skolem dependence function $\theta$ and an assignment $\chi$. Also, assume $s_1 \triangleq \mathtt{tr}(s_0, \emptyset[\alpha \mapsto \chi(\mathtt{x})(s_0)])$ and $s_2 \triangleq \mathtt{tr}(s_0, \emptyset[\alpha \mapsto \theta(\chi)(\mathtt{y})(s_0)])$. Now, we distinguish between two cases.

- $\mathtt{p} \in \mathsf{L}(s_1)$ iff $\mathtt{p} \in \mathsf{L}(s_2)$. In this case, we can easily observe that $\mathcal{G}, \theta(\chi), s_0 \not\models \psi_1$ and consequently, by Theorem 2.2.2 on the preceding page, it holds that $\mathcal{G}, \theta(\chi), s_0 \not\models_{\mathrm{B}} \psi_1 \wedge \psi_2$. So, we are done.

- $\mathtt{p} \in \mathsf{L}(s_1)$ iff $\mathtt{p} \notin \mathsf{L}(s_2)$. If $\mathcal{G}, \theta(\chi), s_0 \not\models \psi_2$ then, by Theorem 2.2.2 on the facing page, it holds that $\mathcal{G}, \theta(\chi), s_0 \not\models_{\mathrm{B}} \psi_1 \wedge \psi_2$. So, we are done. Otherwise, let $s_3 \triangleq \mathtt{tr}(s_1, \emptyset[\alpha \mapsto \chi(\mathtt{x})(s_0 \cdot s_1)])$ and $s_4 \triangleq \mathtt{tr}(s_2, \emptyset[\alpha \mapsto \theta(\chi)(\mathtt{y})(s_0 \cdot s_2)])$. Then, it holds that $\mathtt{p} \in \mathsf{L}(s_3)$ iff $\mathtt{p} \in \mathsf{L}(s_4)$. Now, consider a new assignment $\chi' \in \mathrm{Asg}(\{\mathtt{x}\}, s_0)$ such that $\chi'(x)(s_0 \cdot s_2) = \chi(x)(s_0 \cdot s_2)$ and $\mathtt{p} \in \mathsf{L}({s_3}')$ iff $\mathtt{p} \notin \mathsf{L}(s_4)$, where ${s_3}' \triangleq \mathtt{tr}(s_1, \emptyset[\alpha \mapsto \chi'(\mathtt{x})(s_0 \cdot s_1)])$. Observe that the existence of such an assignment, with particular reference to the second condition, is ensured by the fact that $\mathcal{G} \models \varphi_2$. At this point, due to the behavioral of the Skolem dependence function $\theta$, we have that $\theta(\chi')(\mathtt{y})(s_0 \cdot s_2) = \theta(\chi)(\mathtt{y})(s_0 \cdot s_2)$. Consequently, it holds that $s_4 = \mathtt{tr}(s_2, \emptyset[\alpha \mapsto \theta(\chi')(\mathtt{y})(s_0 \cdot s_2)])$. Thus, $\mathcal{G}, \theta(\chi'), s_0 \not\models \psi_2$, which implies, by Theorem 2.2.2 on the preceding page, that $\mathcal{G}, \theta(\chi'), s_0 \not\models_{\mathrm{B}} \psi_1 \wedge \psi_2$. So, we are done.

Thus, the thesis of the theorem holds. □

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

**Remark 2.2.3 (Kinds of Non-Behavioral)** *It is worth remarking that the kind of non-behavioral of the sentence $\varphi$ shown in the above theorem can be called* essential*, i.e., it cannot be eliminated, due to the fact that $\varphi$ is satisfiable but not behavioral satisfiable. However, there are different sentences, such as the conjunct $\varphi_1$ in $\varphi$, having both models on which they are behaviorally satisfiable and models, like the* CGS *$\mathcal{G}_{Rdc}$, on which they are only non-behaviorally satisfiable. Such a kind of non-behavioral can be called* non-essential*, since it can be eliminated by a suitable choice of the underlying model. Note that a similar reasoning can be done for the dual concept of behavioral, which we call* essential *if all models satisfying a given sentence behaviorally satisfy it as well.*

Before continuing, we want to show the reason why we have redefined the concepts of implication and equivalence in the context of behavioral semantics. Consider the SL[BG, 1-AG, 2-VAR, 1-ALT] sentence $\varphi_1$ used in Theorem 2.2.3 on page 36 of SL[BG] non-behavioral result. It is not hard to see that it is equivalent to the SL[1G, 1-AG, 1-VAR, 0-ALT] $\varphi' \triangleq (\langle\langle \mathtt{x} \rangle\rangle(\alpha, \mathtt{x})\psi_1 \leftrightarrow \langle\langle \mathtt{x} \rangle\rangle(\alpha, \mathtt{x})\psi_2) \wedge (\langle\langle \mathtt{x} \rangle\rangle(\alpha, \mathtt{x})\psi_3 \leftrightarrow \langle\langle \mathtt{x} \rangle\rangle(\alpha, \mathtt{x})\psi_4)$, where $\psi_1 \triangleq \mathtt{X}(\mathtt{p} \wedge \mathtt{Xp})$, $\psi_2 \triangleq \mathtt{X}(\neg\mathtt{p} \wedge \mathtt{Xp})$, $\psi_3 \triangleq \mathtt{X}(\mathtt{p} \wedge \mathtt{X}\neg\mathtt{p})$, and $\psi_4 \triangleq \mathtt{X}(\neg\mathtt{p} \wedge \mathtt{X}\neg\mathtt{p})$. Note that $\varphi'$ is in turn equivalent to the CTL$^\star$ formula $(\mathtt{E}\psi_1 \leftrightarrow \mathtt{E}\psi_2) \wedge (\mathtt{E}\psi_3 \leftrightarrow \mathtt{E}\psi_4)$. However, $\varphi_1$ and $\varphi'$ are not behaviorally equivalent, since we have that $\mathcal{G}_{Rdc} \not\models_{\mathrm{B}} \varphi_1$ but $\mathcal{G}_{Rdc} \models_{\mathrm{B}} \varphi'$, where $\mathcal{G}_{Rdc}$ is the CGS of Figure 1.8 on page 17.

At this point, we can proceed with the proof of the behavioral satisfiability for SL[1G]. It is important to note that there is no gap, in our knowledge, between the logics that are behaviorally satisfiable and those that are not, since the fragment SL[BG, 1-AG, 2-VAR, 1-ALT] used in the previous theorem cannot be further reduced, due to the fact that otherwise it collapses into SL[1G]. Before starting, we have to describe some notation regarding classic two-player games on infinite words [PP04], which are used here as a technical tool. Note that we introduce the names of scheme and match in place of the more usual strategy and play, in order to avoid confusion between the concepts related to a CGS and those related to the tool.

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

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

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

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

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

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

As first step towards the proof of the behavioral of $\mathrm{SL}[1\mathrm{G}]$, we have to give a construction of a two-player game, based on an a priori chosen CGS, in which the players are explicitly viewed one as a Skolem dependence function and the other as a valuation, both over actions. This construction results to be a deep technical evolution of the proof method used for the dualization of alternating automata on infinite objects [MS87].

**Definition 2.2.5 (Dependence-vs-Valuation Game)** *Let $\mathcal{G}$ be a* CGS*, $s \in \mathrm{St}$ one of its states, $\mathrm{P} \subseteq \mathrm{Pth}(s)$ a set of paths, $\wp \in \mathrm{Qnt}(\mathrm{V})$ a quantification prefix over a set $\mathrm{V} \subseteq \mathrm{Vr}$ of variables, and $\flat \in \mathrm{Bn}(\mathrm{V})$ a binding. Then, the dependence-vs-valuation game for $\mathcal{G}$ in $s$ over $\mathrm{P}$ w.r.t. $\wp$ and $\flat$ is the* TPG $\mathcal{H}(\mathcal{G}, s, \mathrm{P}, \wp, \flat) \triangleq \langle \mathcal{A}(\mathcal{G}, s, \wp, \flat), \mathrm{P}\rangle$*, where the* TPA $\mathcal{A}(\mathcal{G}, s, \wp, \flat) \triangleq \langle \mathrm{St}, \mathrm{St} \times \mathrm{SDF}_{\mathrm{Ac}}(\wp), Ed, s\rangle$ *has the edge relations defined as follows:*

- $E_e \triangleq \{(t, (t, \theta)) : t \in \mathrm{St} \wedge \theta \in \mathrm{SDF}_{\mathrm{Ac}}(\wp)\}$*;*
- $E_o \triangleq \{((t, \theta), \mathsf{tr}(t, \theta(\mathsf{v}) \circ \mathsf{bnd}_\flat)) : t \in \mathrm{St} \wedge \theta \in \mathrm{SDF}_{\mathrm{Ac}}(\wp) \wedge \mathsf{v} \in \mathrm{Val}_{\mathrm{Ac}}([\![\wp]\!])\}$ [4].

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

**Lemma 2.2.2 (Dependence-vs-Valuation Duality)** *Let $\mathcal{G}$ be a* CGS*, $s \in \mathrm{St}$ one of its states, $\mathrm{P} \subseteq \mathrm{Pth}(s)$ a set of paths, $\wp \in \mathrm{Qnt}(\mathrm{V})$ a quantification prefix over a set $\mathrm{V} \subseteq \mathrm{Vr}$ of variables, and $\flat \in \mathrm{Bn}(\mathrm{V})$ a binding. Then, player even wins the* TPG $\mathcal{H}(\mathcal{G}, s, \mathrm{P}, \wp, \flat)$ *iff player odd wins the dual* TPG $\mathcal{H}(\mathcal{G}, s, \mathrm{Pth}(s) \setminus \mathrm{P}, \overline{\wp}, \flat)$.

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

**Definition 2.2.6 (Encasement)** *Let $\mathcal{G}$ be a* CGS*, $s \in \mathrm{St}$ one of its states, $\mathrm{P} \subseteq \mathrm{Pth}(s)$ a set of paths, $\wp \in \mathrm{Qnt}(\mathrm{V})$ a quantification prefix over a set $\mathrm{V} \subseteq \mathrm{Vr}$ of variables, and $\flat \in \mathrm{Bn}(\mathrm{V})$ a binding. Then, $\mathrm{P}$ is an* encasement *w.r.t. $\wp$ and $\flat$ if there exists a behavioral Skolem dependence function $\theta \in \mathrm{BSDF}_{\mathrm{Str}(s)}(\wp)$ such that, for all assignments $\chi \in \mathrm{Asg}([\![\wp]\!], s)$, it holds that $\mathsf{play}(\theta(\chi) \circ \mathsf{bnd}_\flat, s) \in \mathrm{P}$.*

In the next lemma, we give the second of the two crucial steps in our behavioral proof. In particular, we are able to show a one-to-one relationship between the winning in the dependence-vs-valuation game of player even and the verification of the encasement property of the associated winning set. Moreover, in the case that the latter is a Borelian set, by using Martin's Determinacy Theorem [Mar75], we obtain a complete characterization of the winning concept by means of that of encasements.

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

**Lemma 2.2.3 (Encasement Characterization)** *Let $\mathcal{G}$ be a* CGS*, $s \in \mathrm{St}$ one of its states, $\mathrm{P} \subseteq \mathrm{Pth}(s)$ a set of paths, $\wp \in \mathrm{Qnt}(\mathrm{V})$ a quantification prefix over a set $\mathrm{V} \subseteq \mathrm{Vr}$ of variables, and $\flat \in \mathrm{Bn}(\mathrm{V})$ a binding. Then, the following hold:*

*(i) player even wins $\mathcal{H}(\mathcal{G}, s, \mathrm{P}, \wp, \flat)$ iff $\mathrm{P}$ is an encasement w.r.t. $\wp$ and $\flat$;*

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

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

Finally, we have all technical tools useful to prove the behavioral of the satisfiability for $\mathrm{SL}[1\mathrm{G}]$. Intuitively, we describe a bidirectional reduction of the problem of interest to the verification of the winning in the dependence-vs-valuation game. The idea behind this construction resides in the strong similarity between the statement of Corollary 2.2.1 on page 33 of SL strategy quantification and the definition of the winning condition in a two-player game. Indeed, on one hand, we say that a sentence is satisfiable iff "there exists a Skolem dependence function such that, for all all assignments, it holds that ...". On the other hand, we say that player even wins a game iff "there exists an even scheme such that, for all odd schemes, it holds that ...". In particular, for the $\mathrm{SL}[1\mathrm{G}]$ fragment, we can resolve the gap between these two formulations, by using the concept of behavioral quantification.

**Theorem 2.2.4 (**$\mathrm{SL}[1\mathrm{G}]$ **Behavioral)** *Let $\mathcal{G}$ be a* CGS*, $\varphi$ an* $\mathrm{SL}[1\mathrm{G}]$ *formula, $s \in \mathrm{St}$ a state, and $\chi \in \mathrm{Asg}(s)$ an $s$-total assignment with $\mathsf{free}(\varphi) \subseteq \mathsf{dom}(\chi)$. Then, it holds that $\mathcal{G}, \chi, s \models \varphi$ iff $\mathcal{G}, \chi, s \models_{\mathrm{B}} \varphi$.*

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

*[If].* The proof of this direction is practically the same of the one used in Theorem 2.2.2 on page 36 of $\mathrm{SL}[\mathrm{NG}]$ behavioral coherence. So, we omit to report it here.

*[Only if].* Assume that $\mathcal{G}, \varnothing, s \models \wp\flat\psi$. Then, it is easy to see that, for all behavioral Skolem dependence functions $\theta \in \mathrm{BSDF}_{\mathrm{Str}(s)}(\overline{\wp})$, there is an assignment $\chi \in \mathrm{Asg}([\![\overline{\wp}]\!], s)$ such that $\mathcal{G}, \theta(\chi) \circ \mathsf{bnd}_{\flat}, s \models \psi$. Indeed, suppose by contradiction that there exists a behavioral Skolem dependence function $\theta \in \mathrm{BSDF}_{\mathrm{Str}(s)}(\overline{\wp})$ such that, for all assignments $\chi \in \mathrm{Asg}([\![\overline{\wp}]\!], s)$, it holds that $\mathcal{G}, \theta(\chi) \circ \mathsf{bnd}_{\flat}, s \not\models \psi$, i.e., $\mathcal{G}, \theta(\chi) \circ \mathsf{bnd}_{\flat}, s \models \neg\psi$, and so $\mathcal{G}, \theta(\chi), s \models \flat\neg\psi$. Then, by Corollary 2.2.1 on page 33 of SL strategy quantification, we have that $\mathcal{G}, \varnothing, s \models \overline{\wp}\flat\neg\psi$, i.e., $\mathcal{G}, \varnothing, s \models \neg\wp\flat\psi$, and so $\mathcal{G}, \varnothing, s \not\models \wp\flat\psi$, which is impossible.

Now, let $\mathrm{P} \triangleq \{\mathsf{play}(\chi, s) \in \mathrm{Pth}(s) : \chi \in \mathrm{Asg}(\mathrm{Ag}, s) \wedge \mathcal{G}, \chi, s \not\models \psi\}$. Then, it is evident that, for all behavioral Skolem dependence functions $\theta \in \mathrm{BSDF}_{\mathrm{Str}(s)}(\overline{\wp})$, there is an assignment $\chi \in \mathrm{Asg}([\![\overline{\wp}]\!], s)$ such that $\mathsf{play}(\theta(\chi) \circ \mathsf{bnd}_{\flat}, s) \notin \mathrm{P}$.

At this point, by Definition 2.2.6 on the previous page of encasement, it is clear that $\mathrm{P}$ is not an encasement w.r.t. $\overline{\wp}$ and $\flat$. Moreover, since $\psi$ describes a regular language, the derived set $\mathrm{P}$ is Borelian [PP04]. Consequently, by Item iii of Lemma 2.2.3 of encasement characterization,

we have that player odd wins the TPG $\mathcal{H}(\mathcal{G}, s, \mathrm{P}, \overline{\wp}, \flat)$. Thus, by Lemma 2.2.2 on page 39 of dependence-vs-valuation duality, player even wins the dual TPG $\mathcal{H}(\mathcal{G}, s, \mathrm{Pth}(s) \setminus \mathrm{P}, \wp, \flat)$. Hence, by Item i of Lemma 2.2.3 on the facing page, we have that $\mathrm{Pth}(s) \setminus \mathrm{P}$ is an encasement w.r.t. $\wp$ and $\flat$. Finally, again by Definition 2.2.6 on page 39, there exists an behavioral Skolem dependence function $\theta \in \mathrm{BSDF}_{\mathrm{Str}(s)}(\wp)$ such that, for all assignments $\chi \in \mathrm{Asg}([\![\wp]\!], s)$, it holds that $\mathsf{play}(\theta(\chi) \circ \mathsf{bnd}_\flat, s) \in \mathrm{Pth}(s) \setminus \mathrm{P}$.

Now, it is immediate to observe that $\mathrm{Pth}(s) \setminus \mathrm{P} = \{\mathsf{play}(\chi, s) \in \mathrm{Pth}(s) : \chi \in \mathrm{Asg}(\mathrm{Ag}, s) \wedge \mathcal{G}, \chi, s \models \psi\}$. So, by the inductive hypothesis, we have that $\mathrm{Pth}(s) \setminus \mathrm{P} = \{\mathsf{play}(\chi, s) \in \mathrm{Pth}(s) : \chi \in \mathrm{Asg}(\mathrm{Ag}, s) \wedge \mathcal{G}, \chi, s \models_{\mathrm{B}} \psi\}$, from which we derive that there exists a behavioral Skolem dependence function $\theta \in \mathrm{BSDF}_{\mathrm{Str}(s)}(\wp)$ such that, for all assignments $\chi \in \mathrm{Asg}([\![\wp]\!], s)$, it holds that $\mathcal{G}, \theta(\chi) \circ \mathsf{bnd}_\flat, s \models_{\mathrm{B}} \psi$. Consequently, by Definition 2.2.4 on page 35 of SL[NG] behavioral semantics, we have that $\mathcal{G}, \varnothing, s \models_{\mathrm{B}} \wp\flat\psi$. □

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

**Corollary 2.2.2 (**SL[1G] **Behavioral)** *Let $\mathcal{G}$ be a* CGS *and $\varphi$ an* SL[1G] *sentence. Then, $\mathcal{G} \models \varphi$ iff $\mathcal{G} \models_{\mathrm{B}} \varphi$.*

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

## 2.3 Model Checking

In this section, we study the model-checking problem for SL and show that, in general, it is non-elementarily decidable, while, in the particular case of SL[1G] sentences, it is just 2EXPTIME-COMPLETE, as for ATL$^\star$. For the algorithmic procedures, we follow an *automata-theoretic approach* [KVW00], reducing the decision problem for the logics to the emptiness problem of an automaton. In particular, we use a bottom-up technique through which we recursively label each state of the CGS of interest by all principal subsentences of the specification that are satisfied on it, starting from the innermost subsentences and terminating with the sentence under exam. In this way, at a given step of the recursion, since the satisfaction of all subsentences of the given principal sentence has already been determined, we can assume that the matrix of the latter is only composed by Boolean combinations and nesting of goals whose temporal part is simply LTL. The procedure we propose here extends that used for ATL$^\star$ in [AHK02] by means of a richer structure of the automata involved in.

The rest of this section is organized as follows. In Subsection 2.3.1, we recall the definition of alternating parity tree automata. Then, in Subsection 2.3.2, we build an automaton accepting a tree encoding of a CGS iff this satisfies the formula of interest, which is used to prove the main result about SL and SL[NG] model checking. Finally, in Subsection 2.3.3, we refine the previous result to obtain an elementary decision procedure for SL[1G].

### 2.3.1 Alternating tree automata

*Nondeterministic tree automata* are a generalization to infinite trees of the classical *nondeterministic word automata* on infinite words (see [Tho90], for an introduction). *Alternating tree automata* are a further generalization of nondeterministic tree automata [MS87]. Intuitively, on visiting a node of the input tree, while the latter sends exactly one copy of itself to each of the successors of the node, the former can send several own copies to the same successor. Here we use, in particular, *alternating parity tree automata*, which are alternating tree automata along with a *parity acceptance condition* (see [GTW02], for a survey).

We now give the formal definition of alternating tree automata.

**Definition 2.3.1 (Alternating Tree Automata)** *An* alternating tree automaton *(*ATA*, for short) is a tuple* $\mathcal{A} \triangleq \langle \mathrm{Sm}, \Delta, \mathrm{Q}, \delta, q_0, \aleph \rangle$*, where* $\Sigma$*,* Dir*, and* Q *are, respectively, nonempty finite sets of* input symbols*,* directions*, and* states*,* $q_0 \in \mathrm{Q}$ *is an* initial state*,* $\aleph$ *is an* acceptance condition *to be defined later, and* $\delta : \mathrm{Q} \times \Sigma \to \mathtt{B}^+(\mathrm{Dir} \times \mathrm{Q})$ *is an* alternating transition function *that maps each pair of states and input symbols to a positive Boolean combination on the set of propositions of the form* $(d, q) \in \mathrm{Dir} \times \mathrm{Q}$*, a.k.a.* moves*.*

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

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

**Definition 2.3.2 (ATA Run)** *A* run *of an* ATA $\mathcal{A} = \langle \mathrm{Sm}, \Delta, \mathrm{Q}, \delta, q_0, \aleph \rangle$ *on a* $\Sigma$*-labeled* Dir*-tree* $\mathcal{T} = \langle \mathrm{T}, \mathsf{v} \rangle$ *is a* $(\mathrm{Dir} \times \mathrm{Q})$*-tree* R *such that, for all nodes* $x \in \mathrm{R}$*, where* $x = \prod_{i=1}^{n} (d_i, q_i)$ *and* $y \triangleq \prod_{i=1}^{n} d_i$ *with* $n \in [0, \omega[$*, it holds that* (i) $y \in \mathrm{T}$ *and* (ii)*, there is a set of moves* $\mathrm{S} \subseteq \mathrm{Dir} \times \mathrm{Q}$ *with* $\mathrm{S} \models \delta(q_n, \mathsf{v}(y))$ *such that* $x \cdot (d, q) \in \mathrm{R}$*, for all* $(d, q) \in \mathrm{S}$*.*

In the following, we consider ATAs along with the *parity acceptance condition* (APT, for short) $\aleph \triangleq (\mathrm{F}_1, \ldots, \mathrm{F}_k) \in (2^{\mathrm{Q}})^+$ with $\mathrm{F}_1 \subseteq \ldots \subseteq \mathrm{F}_k = \mathrm{Q}$ (see [KVW00], for more). The number $k$ of sets in the tuple $\aleph$ is called the *index* of the automaton. We also consider ATAs with the *co-Büchi acceptance condition* (ACT, for short) that is the special parity condition with index 2.

Let R be a run of an ATA $\mathcal{A}$ on a tree $\mathcal{T}$ and $w$ one of its branches. Then, by $\mathsf{inf}(w) \triangleq \{q \in \mathrm{Q} : |\{i \in \mathbb{N} : \exists d \in \mathrm{Dir}.(w)_i = (d, q)\}| = \omega\}$ we denote the set of states that occur infinitely often as the second component of the letters along the branch $w$. Moreover, we say that $w$ satisfies the parity acceptance condition $\aleph = (\mathrm{F}_1, \ldots, \mathrm{F}_k)$ if the least index $i \in [1, k]$ for which $\mathsf{inf}(w) \cap \mathrm{F}_i \neq \emptyset$ is even.

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

**Definition 2.3.3 (ATA Acceptance)** *An* ATA $\mathcal{A} = \langle \mathrm{Sm}, \Delta, \mathrm{Q}, \delta, q_0, \aleph \rangle$ accepts *a* $\Sigma$*-labeled* Dir*-tree* $\mathcal{T}$ *iff is there exists a run* R *of* $\mathcal{A}$ *on* $\mathcal{T}$ *such that all its infinite branches satisfy the acceptance condition* $\aleph$*.*

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

We finally show a simple but useful result about the APT direction projection. To do this, we first need to introduce an extra notation. Let $f \in \mathtt{B}(\mathrm{P})$ be a Boolean formula on a set of propositions P. Then, by $f^{p/q|p\in\mathrm{P}'}$ we denote the formula in which all occurrences of the propositions $p \in \mathrm{P}' \subseteq \mathrm{P}$ in $f$ are replaced by the proposition $q$ belonging to a possibly different set.

**Theorem 2.3.1 (APT Direction Projection)** *Let $\mathcal{A} \triangleq \langle \Sigma \times \mathrm{Dir}, \mathrm{Dir}, \mathrm{Q}, \delta, q_{\mathrm{o}}, \aleph \rangle$ be an* APT *over a set of $m$ directions with $n$ states and index $k$. Moreover, let $d_{\mathrm{o}} \in \mathrm{Dir}$ be a distinguished direction. Then, there exists an* NPT $\mathcal{N}^{d_{\mathrm{o}}} \triangleq \langle \Sigma, \mathrm{Dir}, \mathrm{Q}', \delta', q_{\mathrm{o}}^{'}, \aleph' \rangle$ *with $m \cdot 2^{\mathrm{O}(k \cdot n \cdot \log n)}$ states and index $\mathrm{O}(k \cdot n \cdot \log n)$ such that, for all $\Sigma$-labeled* Dir*-tree $\mathcal{T} \triangleq \langle \mathrm{T}, \mathsf{v} \rangle$, it holds that $\mathcal{T} \in \mathrm{L}(\mathcal{N}^{d_{\mathrm{o}}})$ iff $\mathcal{T}' \in \mathrm{L}(\mathcal{A})$, where $\mathcal{T}'$ is the $(\Sigma \times \mathrm{Dir})$-labeled* Dir*-tree $\langle \mathrm{T}, \mathsf{v}' \rangle$ such that $\mathsf{v}'(t) \triangleq (\mathsf{v}(t), \mathsf{lst}(d_{\mathrm{o}} \cdot t))$, for all $t \in \mathrm{T}$.*

*Proof.* As first step, we use the well-known nondeterminization procedure for APTs [MS95] in order to transform the APT $\mathcal{A}$ into an equivalent NPT $\mathcal{N} = \langle \Sigma \times \mathrm{Dir}, \mathrm{Dir}, \mathrm{Q}'', \delta'', q_{\mathrm{o}}^{''}, \aleph'' \rangle$ with $2^{\mathrm{O}(k \cdot n \cdot \log n)}$ states and index $k' = \mathrm{O}(k \cdot n \cdot \log n)$. Then, we transform the latter into the new NPT $\mathcal{N}^{d_{\mathrm{o}}} \triangleq \langle \Sigma, \mathrm{Dir}, \mathrm{Q}', \delta', q_{\mathrm{o}}^{'}, \aleph' \rangle$ with $m \cdot 2^{\mathrm{O}(k \cdot n \cdot \log n)}$ states and same index $k'$, where $\mathrm{Q}' \triangleq \mathrm{Q}'' \times \mathrm{Dir}$, $q_{\mathrm{o}}^{'} \triangleq (q_{\mathrm{o}}^{''}, d_{\mathrm{o}})$, $\aleph' \triangleq (\mathrm{F}_1 \times \mathrm{Dir}, \ldots, \mathrm{F}_{k'} \times \mathrm{Dir})$ with $\aleph'' \triangleq (\mathrm{F}_1, \ldots, \mathrm{F}_{k'})$, and $\delta'((q,d),\sigma) \triangleq \delta''(q,(\sigma,d))[(d',q')/(d',(q',d')) \mid (d',q') \in \mathrm{Dir} \times \mathrm{Q}'']$, for all $(q,d) \in \mathrm{Q}'$ and $\sigma \in \Sigma$. Now, it easy to see that $\mathcal{N}^{d_{\mathrm{o}}}$ satisfies the declared statement. □

### 2.3.2 SL Model Checking

A first step towards our construction of an algorithmic procedure for the solution of the SL model-checking problem is to define, for each possible formula $\varphi$, an alternating parity tree automaton $\mathcal{A}_{\varphi}^{\mathcal{G}}$ that recognizes a tree encoding $\mathcal{T}$ of a CGS $\mathcal{G}$, containing the information on an assignment $\chi$ on the free variables/agents of $\varphi$, iff $\mathcal{G}$ is a model of $\varphi$ under $\chi$. The high-level idea at the base of this construction is an evolution and merging of those behind the translations of QPTL and LTL, respectively, into nondeterministic [SVW87] and alternating [MSS88] Büchi automata.

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

**Definition 2.3.4 (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 $\mathrm{V} \subseteq \mathrm{Vr} \cup \mathrm{Ag}$. Then, a $(\mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathrm{V}) \times \mathrm{St}_{\mathcal{G}})$-labeled $\mathrm{St}_{\mathcal{G}}$-tree $\mathcal{T} \triangleq \langle \mathrm{T}, \mathsf{u} \rangle$, where $\mathrm{T} \triangleq \{\rho_{\geq 1} : \rho \in \mathrm{Trk}_{\mathcal{G}}(s)\}$, is an* assignment-state encoding *for $\chi$ if it holds that $\mathsf{u}(t) \triangleq (\widehat{\chi}(s \cdot t), \mathsf{lst}(s \cdot t))$, for all $t \in \mathrm{T}$.*

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

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

**Lemma 2.3.1 (SL Formula Automaton)** *Let $\mathcal{G}$ be a* CGS *and $\varphi$ an* SL *formula. Then, there exists an* APT $\mathcal{A}^{\mathcal{G}}_{\varphi} \triangleq \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathsf{free}(\varphi)) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\varphi}, \delta_{\varphi}, q_{\mathrm{o}\varphi}, \aleph_{\varphi} \rangle$ *such that, for all states $s \in \mathrm{St}_{\mathcal{G}}$ and assignments $\chi \in \mathrm{Asg}_{\mathcal{G}}(\mathsf{free}(\varphi), s)$, it holds that $\mathcal{G}, \chi, s \models \varphi$ iff $\mathcal{T} \in \mathrm{L}(\mathcal{A}^{\mathcal{G}}_{\varphi})$, where $\mathcal{T}$ is the assignment-state encoding for $\chi$.*

*Proof.* The construction of the APT $\mathcal{A}^{\mathcal{G}}_{\varphi}$ is done recursively on the structure of the formula $\varphi$, which w.l.o.g. is supposed to be in *enf*, by using a variation of the transformation, via alternating tree automata, of the S1S and S$k$S logics into nondeterministic Büchi word and tree automata recognizing all models of the formula of interest [Büc62, Rab69].

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

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

- The boolean case $\varphi = \neg\varphi'$ is treated in the classical way, by simply dualizing the automaton $\mathcal{A}^{\mathcal{G}}_{\varphi'} = \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathsf{free}(\varphi')) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\varphi'}, \delta_{\varphi'}, q_{\mathrm{o}\varphi'}, \aleph_{\varphi'} \rangle$ [MS87].

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

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

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

  - $\mathrm{Q}_{\varphi} \triangleq \{q_{\mathrm{o}\varphi}\} \cup \mathrm{Q}_{\varphi'}$, with $q_{\mathrm{o}\varphi} \notin \mathrm{Q}_{\varphi'}$;
  - $\delta_{\varphi}(q_{\mathrm{o}\varphi}, (\mathsf{v}, s)) \triangleq (\mathsf{tr}_{\mathcal{G}}(s, \mathsf{v}_{\restriction \mathrm{Ag}}), q_{\mathrm{o}\varphi'})$, for all $(\mathsf{v}, s) \in \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathsf{free}(\varphi)) \times \mathrm{St}_{\mathcal{G}}$;

– $\delta_{\varphi}(q,(\mathsf{v},s)) \triangleq \delta_{\varphi'}(q,(\mathsf{v}_{\restriction \mathsf{free}(\varphi')},s))$, for all $q \in \mathrm{Q}_{\varphi'}$ and $(\mathsf{v},s) \in \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathsf{free}(\varphi)) \times \mathrm{St}_{\mathcal{G}}$;

– $\aleph_{\varphi} \triangleq (\mathrm{F}_{1\varphi'},\ldots,\mathrm{F}_{k\varphi'} \cup \{q_{0\varphi}\})$, where $\aleph_{\varphi'} \triangleq (\mathrm{F}_{1\varphi'},\ldots,\mathrm{F}_{k\varphi'})$.

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

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

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

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

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

  It is important to observe that the initial state $q_{0\varphi}$ is included in all sets of the parity acceptance condition, in particular in $\mathrm{F}_{1\varphi}$, in order to avoid its regeneration for an infinite number of times.

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

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

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

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

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

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

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

- To handle the case $\varphi = \langle\!\langle x \rangle\!\rangle \varphi'$, assuming that $x \in \mathsf{free}(\varphi')$, we use the operation of existential projection for nondeterministic tree automata. To do this, we have first to nondeterminize the APT $\mathcal{A}^{\mathcal{G}}_{\varphi'}$, by applying the classic transformation [MS95]. In this way, we obtain an equivalent NPT $\mathcal{N}^{\mathcal{G}}_{\varphi'} = \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathsf{free}(\varphi')) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\varphi'}, \delta_{\varphi'}, q_{\mathrm{o}\varphi'}, \aleph_{\varphi'} \rangle$. Now, we make the projection, by defining the new NPT $\mathcal{A}^{\mathcal{G}}_{\varphi} \triangleq \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathsf{free}(\varphi)) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\varphi'}, \delta_{\varphi}, q_{\mathrm{o}\varphi'}, \aleph_{\varphi'} \rangle$ where $\delta_{\varphi}(q, (\mathsf{v}, s)) \triangleq \bigvee_{c \in \mathrm{Ac}_{\mathcal{G}}} \delta_{\varphi'}(q, (\mathsf{v}_{x \mapsto c}, s))$, for all $q \in \mathrm{Q}_{\varphi'}$ and $(\mathsf{v}, s) \in \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathsf{free}(\varphi)) \times \mathrm{St}_{\mathcal{G}}$.

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

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

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

**Theorem 2.3.2 (SL Sentence Automaton)** *Let $\mathcal{G}$ be a* CGS*, $s \in \mathrm{St}_{\mathcal{G}}$ one of its states, and $\varphi$ an* SL *sentence. Then, there exists an* NPT $\mathcal{N}^{\mathcal{G},s}_{\varphi}$ *such that $\mathcal{G}, \varnothing, s \models \varphi$ iff $\mathrm{L}(\mathcal{N}^{\mathcal{G},s}_{\varphi}) \neq \emptyset$.*

*Proof.* To construct the NPT $\mathcal{N}^{\mathcal{G},s}_{\varphi}$ we apply Theorem 2.3.1 on page 43 of APT direction projection with distinguished direction $s$ to the APT $\mathcal{A}^{\mathcal{G}}_{\varphi}$ derived by Lemma 2.3.1 on page 44 of SL formula automaton. In this way, we can ensure that the state labeling of nodes of the assignment-state encoding is coherent with the node itself. Observe that, since $\varphi$ is a sentence, we have that $\mathsf{free}(\varphi) = \emptyset$, and so, the unique assignment-state encoding of interest is that related to the empty assignment $\varnothing$.

*[Only if].* Suppose that $\mathcal{G}, \varnothing, s \models \varphi$. Then, by Lemma 2.3.1 on page 44, we have that $\mathcal{T} \in \mathrm{L}(\mathcal{A}^{\mathcal{G}}_{\varphi})$, where $\mathcal{T}$ is the dependence-state encoding for $\varnothing$. Hence, by Theorem 2.3.1 on page 43, it holds that $\mathrm{L}(\mathcal{N}^{\mathcal{G},s}_{\varphi}) \neq \emptyset$.

*[If].* Suppose that $\mathrm{L}(\mathcal{N}^{\mathcal{G},s}_{\varphi}) \neq \emptyset$. Then, by Theorem 2.3.1 on page 43, there exists an $(\{\varnothing\} \times \mathrm{St}_{\mathcal{G}})$-labeled $\mathrm{St}_{\mathcal{G}}$-tree $\mathcal{T}$ such that $\mathcal{T} \in \mathrm{L}(\mathcal{A}^{\mathcal{G}}_{\varphi})$. Now, it is immediate to see that $\mathcal{T}$ is the assignment-state encoding for $\varnothing$. Hence, by Lemma 2.3.1 on page 44, we have that $\mathcal{G}, \varnothing, s \models \varphi$. □

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

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

**Theorem 2.3.3** (SL **Model Checking)** *The model-checking problem for* SL *is* PTIME-COMPLETE *w.r.t. the size of the model and* NONELEMENTARY *w.r.t. the size of the specification.*

*Proof.* By Theorem 2.3.2 on the preceding page of SL sentence automaton, to verify that $\mathcal{G}, \varnothing, s \models \varphi$, we simply calculate the emptiness of the NPT $\mathcal{N}_{\varphi}^{\mathcal{G},s}$ having $|\mathrm{St}_{\mathcal{G}}| \cdot (2 \uparrow\uparrow_m m)$ states and index $2 \uparrow\uparrow_m m$, where $m = \mathrm{O}(\mathsf{lng}(\varphi) \cdot \log \mathsf{lng}(\varphi))$. It is well-known that the emptiness problem for such a kind of automaton with $n$ states and index $h$ is solvable in time $\mathrm{O}(n^h)$ [KV98]. Thus, we get that the time complexity of checking whether $\mathcal{G}, \varnothing, s \models \varphi$ is $|\mathrm{St}_{\mathcal{G}}|^{2 \uparrow\uparrow_m m}$. Hence, the membership of the model-checking problem for SL in PTIME w.r.t. the size of the model and NONELEMENTARY w.r.t. the size of the specification directly follows. Finally, by getting the relative lower bound on the model from the same problem for ATL$^\star$ [AHK02], the claim is proved. □

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

**Theorem 2.3.4** (SL[NG] **Model Checking)** *The model-checking problem for* SL[NG] *is* PTIME-COMPLETE *w.r.t. the size of the model and* $(k+1)$-EXPTIME *w.r.t. the maximum alternation* $k$ *of the specification.*

*Proof.* By Theorem 2.3.2 on the facing page of SL sentence automaton, to verify that $\mathcal{G}, \varnothing, s \models \wp\psi$, where $\wp\psi$ is an SL[NG] principal sentence without proper subsentences, we can simply calculate the emptiness of the NPT $\mathcal{N}_{\wp\psi}^{\mathcal{G},s}$ having $|\mathrm{St}_{\mathcal{G}}| \cdot (2 \uparrow\uparrow_m k)$ states and index $2 \uparrow\uparrow_m k$, where $m = \mathrm{O}(\mathsf{lng}(\psi) \cdot \log \mathsf{lng}(\psi))$ and $k = \mathsf{alt}(\wp\psi)$. Thus, we get that the time complexity of checking whether $\mathcal{G}, \varnothing, s \models \wp\psi$ is $|\mathrm{St}_{\mathcal{G}}|^{2 \uparrow\uparrow_m k}$. At this point, since we have to do this verification for each possible state $s \in \mathrm{St}_{\mathcal{G}}$ and principal subsentence $\wp\psi \in \mathsf{psnt}(\varphi)$ of the whole SL[NG] specification $\varphi$, we derive that the bottom-up model-checking procedure requires time $|\mathrm{St}_{\mathcal{G}}|^{2 \uparrow\uparrow_{\mathsf{lng}(\varphi)} k}$, where $k = \mathsf{max}\{\mathsf{alt}(\wp\psi) : \wp\psi \in \mathsf{psnt}(\varphi)\}$. Hence, the membership of the model-checking problem for SL in PTIME w.r.t. the size of the model and $(k+1)$-EXPTIME w.r.t. the maximum alternation $k$ of the specification directly follows. Finally, by getting the relative lower bound on the model from the same problem for ATL$^\star$ [AHK02], the thesis is proved. □

### 2.3.3 SL[1G] Model Checking

We now show how the concept of behavioral of Skolem dependence functions over strategies can be used to enormously reduce the complexity of the model-checking procedure for the SL[1G] fragment. The idea behind our approach is to avoid the use of projections used to

handle the strategy quantifications, by reducing them to action quantifications, which can be managed locally on each state of the model without a tower of exponential blow-ups.

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

**Lemma 2.3.2 (SL[1G] Goal Automaton)** *Let $\mathcal{G}$ be a* CGS *and $\flat\psi$ an* SL[1G] *goal without principal subsentences. Then, there exists an* UCT $\mathcal{U}^{\mathcal{G}}_{\flat\psi} \triangleq \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathsf{free}(\flat\psi)) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\flat\psi}, \delta_{\flat\psi}, q_{\flat\psi}, \aleph_{\flat\psi}\rangle$ *such that, for all states $s \in \mathrm{St}_{\mathcal{G}}$ and assignments $\chi \in \mathrm{Asg}_{\mathcal{G}}(\mathsf{free}(\flat\psi), s)$, it holds that $\mathcal{G}, \chi, s \models \flat\psi$ iff $\mathcal{T} \in \mathrm{L}(\mathcal{U}^{\mathcal{G}}_{\flat\psi})$, where $\mathcal{T}$ is the assignment-state encoding for $\chi$.*

*Proof.* A first step in the construction of the UCT $\mathcal{U}^{\mathcal{G}}_{\flat\psi}$ is to consider the UCW $\mathcal{U}_{\psi} \triangleq \langle 2^{\mathrm{AP}}, \mathrm{Q}_{\psi}, \delta_{\psi}, \mathrm{Q}_{\mathrm{o}\psi}, \aleph_{\psi}\rangle$ obtained by dualizing the NBW resulting from the application of the classic Vardi-Wolper construction to the LTL formula $\neg\psi$ [VW86]. Observe that $\mathrm{L}(\mathcal{U}_{\psi}) = \mathrm{L}(\psi)$, i.e., $\mathcal{U}_{\psi}$ recognizes all infinite words on the alphabet $2^{\mathrm{AP}}$ that satisfy the LTL formula $\psi$. Then, define the components of $\mathcal{U}^{\mathcal{G}}_{\flat\psi} \triangleq \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathsf{free}(\flat\psi)) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\flat\psi}, \delta_{\flat\psi}, q_{\mathrm{o}\flat\psi}, \aleph_{\flat\psi}\rangle$ as follows:

- $\mathrm{Q}_{\flat\psi} \triangleq \{q_{\mathrm{o}\flat\psi}\} \cup \mathrm{Q}_{\psi}$, with $q_{\mathrm{o}\flat\psi} \notin \mathrm{Q}_{\psi}$;
- $\delta_{\flat\psi}(q_{\mathrm{o}\flat\psi}, (\mathsf{v}, s)) \triangleq \bigwedge_{q \in \mathrm{Q}_{\mathrm{o}\psi}} \delta_{\flat\psi}(q, (\mathsf{v}, s))$, for all $(\mathsf{v}, s) \in \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathsf{free}(\flat\psi)) \times \mathrm{St}_{\mathcal{G}}$;
- $\delta_{\flat\psi}(q, (\mathsf{v}, s)) \triangleq \bigwedge_{q' \in \delta_{\psi}(q, \mathsf{L}_{\mathcal{G}}(s))} (\mathsf{tr}_{\mathcal{G}}(s, \mathsf{v} \circ \mathsf{bnd}_{\flat}), q')$, for all $q \in \mathrm{Q}_{\psi}$ and $(\mathsf{v}, s) \in \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathsf{free}(\flat\psi)) \times \mathrm{St}_{\mathcal{G}}$;
- $\aleph_{\flat\psi} \triangleq \aleph_{\psi}$.

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

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

**Definition 2.3.5 (Behavioral Dependence-State Encoding)** *Let $\mathcal{G}$ be a* CGS*, $s \in \mathrm{St}_{\mathcal{G}}$ one of its states, and $\theta \in \mathrm{BSDF}_{\mathrm{Str}_{\mathcal{G}}(s)}(\wp)$ a behavioral Skolem dependence function over strategies for a quantification prefix $\wp \in \mathrm{Qnt}(\mathrm{V})$ over the set $\mathrm{V} \subseteq \mathrm{Vr}$. Then, a $(\mathrm{SDF}_{\mathrm{Ac}_{\mathcal{G}}}(\wp) \times \mathrm{St}_{\mathcal{G}})$-labeled $\mathrm{St}_{\mathcal{G}}$-tree $\mathcal{T} \triangleq \langle \mathrm{T}, \mathsf{u}\rangle$, where $\mathrm{T} \triangleq \{\rho_{\geq 1} : \rho \in \mathrm{Trk}_{\mathcal{G}}(s)\}$, is a* behavioral dependence-state encoding *for $\theta$ if it holds that $\mathsf{u}(t) \triangleq (\widetilde{\theta}(s \cdot t), \mathsf{lst}(s \cdot t))$, for all $t \in \mathrm{T}$.*

Observe that there exists a unique behavioral dependence-state encoding for each behavioral Skolem dependence function over strategies.

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

**Lemma 2.3.3** (SL[1G] **Sentence Automaton)** *Let $\mathcal{G}$ be a* CGS *and $\wp\flat\psi$ an* SL[1G] *principal sentence without principal subsentences. Then, there exists a* UCT $\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi} \triangleq \langle \mathrm{SDF}_{\mathrm{Ac}_{\mathcal{G}}}(\wp) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\wp\flat\psi}, \delta_{\wp\flat\psi}, q_{\mathrm{o}\wp\flat\psi}, \aleph_{\wp\flat\psi}\rangle$ *such that, for all states $s \in \mathrm{St}_{\mathcal{G}}$ and behavioral Skolem dependence functions over strategies $\theta \in \mathrm{BSDF}_{\mathrm{Str}_{\mathcal{G}}(s)}(\wp)$, it holds that $\mathcal{G}, \theta(\chi), s \models_{\mathrm{B}} \flat\psi$, for all $\chi \in \mathrm{Asg}_{\mathcal{G}}([\![\wp]\!], s)$, iff $\mathcal{T} \in \mathrm{L}(\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi})$, where $\mathcal{T}$ is the behavioral dependence-state encoding for $\theta$.*

*Proof.* By Lemma 2.3.2 on the preceding page of SL[1G] goal automaton, there is an UCT $\mathcal{U}^{\mathcal{G}}_{\flat\psi} = \langle \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}(\mathsf{free}(\flat\psi)) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\flat\psi}, \delta_{\flat\psi}, q_{\mathrm{o}\flat\psi}, \aleph_{\flat\psi}\rangle$ such that, for all states $s \in \mathrm{St}_{\mathcal{G}}$ and assignments $\chi \in \mathrm{Asg}_{\mathcal{G}}(\mathsf{free}(\flat\psi), s)$, it holds that $\mathcal{G}, \chi, s \models \flat\psi$ iff $\mathcal{T} \in \mathrm{L}(\mathcal{U}^{\mathcal{G}}_{\flat\psi})$, where $\mathcal{T}$ is the assignment-state encoding for $\chi$.

Now, transform $\mathcal{U}^{\mathcal{G}}_{\flat\psi}$ into the new UCT $\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi} \triangleq \langle \mathrm{SDF}_{\mathrm{Ac}_{\mathcal{G}}}(\wp) \times \mathrm{St}_{\mathcal{G}}, \mathrm{St}_{\mathcal{G}}, \mathrm{Q}_{\wp\flat\psi}, \delta_{\wp\flat\psi}, q_{\mathrm{o}\wp\flat\psi}, \aleph_{\wp\flat\psi}\rangle$, with $\mathrm{Q}_{\wp\flat\psi} \triangleq \mathrm{Q}_{\flat\psi}$, $q_{\mathrm{o}\wp\flat\psi} \triangleq q_{\mathrm{o}\flat\psi}$, and $\aleph_{\wp\flat\psi} \triangleq \aleph_{\flat\psi}$, which is used to handle the quantification prefix $\wp$ atomically, where the transition function is defined as follows: $\delta_{\wp\flat\psi}(q, (\theta, s)) \triangleq \bigwedge_{\mathsf{v} \in \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}([\![\wp]\!])} \delta_{\flat\psi}(q, (\theta(\mathsf{v}), s))$, for all $q \in \mathrm{Q}_{\wp\flat\psi}$ and $(\theta, s) \in \mathrm{SDF}_{\mathrm{Ac}_{\mathcal{G}}}(\wp) \times \mathrm{St}_{\mathcal{G}}$. Intuitively, $\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi}$ reads an action Skolem dependence function $\theta$ on each node of the input tree $\mathcal{T}$ labeled with a state $s$ of $\mathcal{G}$ and simulates the execution of the transition function $\delta_{\flat\psi}(q, (\mathsf{v}, s))$ of $\mathcal{U}^{\mathcal{G}}_{\flat\psi}$, for each possible valuation $\mathsf{v} = \theta(\mathsf{v}')$ on $\mathsf{free}(\flat\psi)$ obtained from $\theta$ by a universal valuation $\mathsf{v}' \in \mathrm{Val}_{\mathrm{Ac}_{\mathcal{G}}}([\![\wp]\!])$. It is important to observe that we cannot move the component set $\mathrm{SDF}_{\mathrm{Ac}_{\mathcal{G}}}(\wp)$ from the input alphabet to the states of $\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi}$, by making a related guessing of the Skolem dependence function $\theta$ in the transition function, since we have to ensure that all states in a given node of the tree $\mathcal{T}$, i.e., in each track of the original model $\mathcal{G}$, make the same choice for $\theta$.

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

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

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

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

**Theorem 2.3.5 (**SL[1G] **Sentence Automaton)** *Let $\mathcal{G}$ be a* CGS*, $s \in \mathrm{St}_{\mathcal{G}}$ one of its states, and $\wp\flat\psi$ an* SL[1G] *principal sentence without principal subsentences. Then, there exists an* NPT $\mathcal{N}^{\mathcal{G},s}_{\wp\flat\psi}$ *such that $\mathcal{G}, \varnothing, s \models \wp\flat\psi$ iff $\mathrm{L}(\mathcal{N}^{\mathcal{G},s}_{\wp\flat\psi}) \neq \emptyset$.*

*Proof.* As in the general case of SL sentence automaton, we have to ensure that the state labeling of nodes of the behavioral dependence-state encoding is coherent with the node itself. To do this, we apply Theorem 2.3.1 on page 43 of APT direction projection with distinguished direction $s$ to the UPT $\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi}$ derived by Lemma 2.3.3 on the previous page of the SL[1G] sentence automaton, thus obtaining the required NPT $\mathcal{N}^{\mathcal{G},s}_{\wp\flat\psi}$.

*[Only if].* Suppose that $\mathcal{G}, \varnothing, s \models \wp\flat\psi$. By Corollary 2.2.2 on page 41 of SL[1G] behavioral, it means that $\mathcal{G}, \varnothing, s \models_{\mathrm{B}} \wp\flat\psi$. Then, by Definition 2.2.4 on page 35 of SL[NG] behavioral semantics, there exists an behavioral Skolem dependence function $\theta \in \mathrm{BSDF}_{\mathrm{Str}_{\mathcal{G}}(s)}(\wp)$ such that $\mathcal{G}, \theta(\chi), s \models_{\mathrm{B}} \flat\psi$, for all $\chi \in \mathrm{Asg}_{\mathcal{G}}([\![\wp]\!], s)$. Thus, by Lemma 2.3.3 on the preceding page, we have that $\mathcal{T} \in \mathrm{L}(\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi})$, where $\mathcal{T}$ is the behavioral dependence-state encoding for $\theta$. Hence, by Theorem 2.3.1 on page 43, it holds that $\mathrm{L}(\mathcal{N}^{\mathcal{G},s}_{\wp\flat\psi}) \neq \emptyset$.

*[If].* Suppose that $\mathrm{L}(\mathcal{N}^{\mathcal{G},s}_{\wp\flat\psi}) \neq \emptyset$. Then, by Theorem 2.3.1 on page 43, there exists an $(\mathrm{SDF}_{\mathrm{Ac}_{\mathcal{G}}}(\wp) \times \mathrm{St}_{\mathcal{G}})$-labeled $\mathrm{St}_{\mathcal{G}}$-tree $\mathcal{T}$ such that $\mathcal{T} \in \mathrm{L}(\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi})$. Now, it is immediate to see that there is an behavioral Skolem dependence function $\theta \in \mathrm{BSDF}_{\mathrm{Str}_{\mathcal{G}}(s)}(\wp)$ for which $\mathcal{T}$ is the behavioral dependence-state encoding. Thus, by Lemma 2.3.3 on the previous page, we have that $\mathcal{G}, \theta(\chi), s \models_{\mathrm{B}} \flat\psi$, for all $\chi \in \mathrm{Asg}_{\mathcal{G}}([\![\wp]\!], s)$. By Definition 2.2.4 on page 35 of SL[NG] behavioral semantics, it holds that $\mathcal{G}, \varnothing, s \models_{\mathrm{B}} \wp\flat\psi$. Hence, by Corollary 2.2.2 on page 41 of SL[1G] behavioral, it means that $\mathcal{G}, \varnothing, s \models \wp\flat\psi$. □

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

**Theorem 2.3.6 (**SL[1G] **Model Checking)** *The model-checking problem for* SL[1G] *is* PTIME-COMPLETE *w.r.t. the size of the model and 2*EXPTIME-COMPLETE *w.r.t. the size of the specification.*

*Proof.* By Theorem 2.3.5 of SL[1G] sentence automaton, to verify that $\mathcal{G}, \varnothing, s \models \wp\flat\psi$, we simply calculate the emptiness of the NPT $\mathcal{N}^{\mathcal{G},s}_{\wp\flat\psi}$. This automaton is obtained by the operation of direction projection on the UCT $\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi}$, which is in turn derived by the UCT $\mathcal{U}^{\mathcal{G}}_{\flat\psi}$. Now, it is easy to see that the number of states of $\mathcal{U}^{\mathcal{G}}_{\flat\psi}$, and consequently of $\mathcal{U}^{\mathcal{G}}_{\wp\flat\psi}$, is $2^{\mathrm{O}(\mathsf{lng}(\psi))}$. So, $\mathcal{N}^{\mathcal{G},s}_{\wp\flat\psi}$ has $|\mathrm{St}_{\mathcal{G}}| \cdot 2^{2^{\mathrm{O}(\mathsf{lng}(\psi))}}$ states and index $2^{\mathrm{O}(\mathsf{lng}(\psi))}$.

The emptiness problem for such a kind of automaton with $n$ states and index $h$ is solvable in time $\mathrm{O}(n^h)$ [KV98]. Thus, we get that the time complexity of checking whether $\mathcal{G}, \varnothing, s \models \wp\flat\psi$ is $|\mathrm{St}_{\mathcal{G}}|^{2^{\mathrm{O}(\mathsf{lng}(\psi))}}$. At this point, since we have to do this verification for each possible state $s \in \mathrm{St}_{\mathcal{G}}$ and principal subsentence $\wp\flat\psi \in \mathsf{psnt}(\varphi)$ of the whole SL[1G] specification $\varphi$, we derive that the whole bottom-up model-checking procedure requires time $|\mathrm{St}_{\mathcal{G}}|^{2^{\mathrm{O}(\mathsf{lng}(\varphi))}}$. Hence, the membership of the model-checking problem for SL[1G] in PTIME w.r.t. the size of the model and 2EXPTIME w.r.t. the size of the specification directly follows. Finally the thesis is proved, by getting the relative lower bounds from the same problem for ATL$^\star$ [AHK02]. □

## 2.4 Satisfiability

As shown in Section 1.3.2, SL does not have the bounded model property and, as a trivial consequence, its satisfiability problem is undecidable. By means of Remark 2.1.2 on page 29, we derive that also the fragment SL[BG] has an undecidable satisfiability problem. On the contrary, it is well-known that the satisfiability problem for ATL$^\star$ is 2EXPTIME-COMPLETE [Sch08]. This gap in complexity between SL and ATL$^\star$ gives naturally rise to the question of which are the inherent properties of ATL$^\star$ that make the problem decidable. Here we answer such question by analyzing the syntactic fragment SL[1G] introduced in Section 2.1. We show that this fragment retains all positive properties of ATL$^\star$, such as the decision-tree model property and the bounded model property, which allows us to show that its satisfiability problem is 2EXPTIME-COMPLETE. A fundamental feature used as a tool to prove the announced properties is again the *behavioral satisfiability*.

### 2.4.1 Tree-model property

The satisfiability procedure we propose in this thesis is based on the use of alternating tree automata. Consequently, we need to establish a kind of *tree model property*, which is based on a special sub-class of CGSs, namely, the *concurrent game-trees* (CGT, for short), whose structure of the underlying graph is a tree.

**Definition 2.4.1 (Concurrent Game Trees)** *A* concurrent game tree *(*CGT*, for short) is a* CGS $\mathcal{T} \triangleq \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}, \mathrm{St}, \mathsf{tr}, \mathsf{ap}, s_{\mathrm{o}} \rangle$*, where* (i) $\mathrm{St} \subseteq \mathrm{Dir}^*$ *is a* Dir*-tree for a given set* Dir *of directions and* (ii) *if* $t \cdot e \in \mathrm{St}$ *then there is a decision* $\delta \in \mathrm{Dc}$ *such that* $\mathsf{tr}(t, \delta) = t \cdot e$*, for all* $t \in \mathrm{St}$ *and* $e \in \mathrm{Dir}$*. Furthermore,* $\mathcal{T}$ *is a* decision tree *(DT, for short) if* (i) $\mathrm{St} = \mathrm{Dc}^*$ *and* (ii) *if* $t \cdot \delta \in \mathrm{St}$ *then* $\mathsf{tr}(t, \delta) = t \cdot \delta$*, for all* $t \in \mathrm{St}$ *and* $\delta \in \mathrm{Dc}$*.*

Intuitively, CGTs are CGSs having a transition relation with a tree shape and DTs have, in addition, the states that uniquely determine the history of the computation leading to them. Observe that, for each non trivial track $\rho$ (resp., path $\pi$), there exists a unique finite (resp., infinite) sequence of decisions $\delta_{\mathrm{o}} \cdot \ldots \cdot \delta_{|\rho|-2} \in \mathrm{Dc}^*$ (resp., $\delta_{\mathrm{o}} \cdot \delta_1 \cdot \ldots \in \mathrm{Dc}^\omega$) such that $\rho_{(i+1)} = \mathsf{tr}(\rho_{(i)}, \delta_i)$ (resp., $\pi_{(i+1)} = \mathsf{tr}(\pi_{(i)}, \delta_i)$), for all $i \in [0, |\rho| - 1[$ (resp., $i \in \mathbb{N}$).

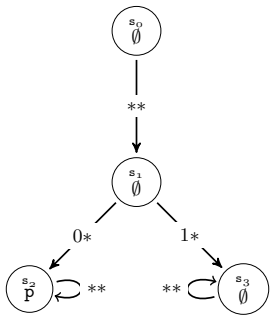



Figure 2.2: A CGS.

We now define a generalization for CGSs of the classic concept of *unwinding* of labeled transition systems, namely the *decision-unwinding* (see Figure 2.2 and Figure 2.3 on the next page, for an example), that allows to show that SL[1G] enjoys the decision-tree model property.

**Definition 2.4.2 (Decision-Unwinding)** *Let* $\mathcal{G} = \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}, \mathrm{St}, \mathsf{tr}, \mathsf{ap}, s_{\mathrm{o}} \rangle$ *be a* CGS*. Then, the* decision-unwinding *of* $\mathcal{G}$ *is the* DT $\mathcal{G}_{DU} \triangleq \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}, \mathrm{Dc}^*, \mathsf{ap}', \mathsf{tr}', \varepsilon \rangle$ *for which there is a surjective function* $\mathsf{unw} : \mathrm{Dc}^* \to \mathrm{St}$ *such that* (i) $\mathsf{unw}(\varepsilon) = s_{\mathrm{o}}$*,* (ii) $\mathsf{unw}(\mathsf{tr}'(t, \delta)) = \mathsf{tr}(\mathsf{unw}(t), \delta)$*, and* (iii) $\mathsf{ap}'(t) = \mathsf{ap}(\mathsf{unw}(t))$*, for all* $t \in \mathrm{Dc}^*$ *and* $\delta \in \mathrm{Dc}$*.*

Observe that, due to its construction, each CGS $\mathcal{G}$ has a unique associated decision-unwinding $\mathcal{G}_{DU}$.

We are now able to prove that SL[1G] satisfies the decision-tree model property.

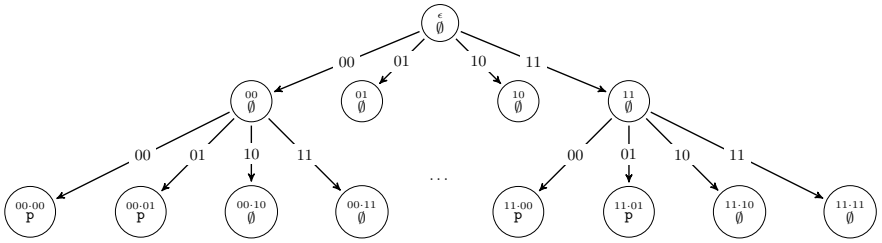



Figure 2.3: Part of the decision-unwinding of the CGS in Figure 2.2.

**Theorem 2.4.1 (**SL[1G] **Decision-Tree Model Property)** *Let $\varphi$ be a satisfiable* SL[1G] *sentence. Then, there exists a* DT $\mathcal{T}$ *such that* $\mathcal{T} \models \varphi$.

*Proof.* The proof proceeds by structural induction on the sentence SL[1G]. For the Boolean combination of principal sentences, the induction is trivial. For the case of a principal sentence $\varphi$ of the form $\wp \flat \psi$, by Theorem 2.2.4 on page 40, we derive that there exists a behavioral Skolem map $\theta \in \mathrm{BSDF}_{\mathrm{Str}_{\mathcal{G}}(s_{\mathrm{o}})}(\wp)$ such that $\mathcal{G} \models_{\theta} \wp \flat \psi$. Furthermore, there exists the adjoint function to tracks $\widetilde{\theta}$ of $\theta$. Now, consider the decision unwinding $\mathcal{T} = \mathcal{G}_{DU}$ of $\mathcal{G}$ and the lifting $\Gamma : \mathrm{Trk}_{\mathcal{T}} \to \mathrm{Trk}_{\mathcal{G}}$ of the unwinding function $\mathsf{unw}$ such that $\Gamma(\rho) = \mathsf{unw}(\rho_{\mathrm{o}}) \cdot \ldots \mathsf{unw}(\rho_{|\rho|-1})$, for all $\rho \in \mathrm{Trk}_{\mathcal{T}}$. At this point, consider the function $\widetilde{\theta'}$ such that $\widetilde{\theta'}(\rho') \triangleq \widetilde{\theta}(\Gamma(\rho'))$, for all $\rho' \in \mathrm{Trk}_{\mathcal{T}}$. Clearly, since $\widetilde{\theta}$ is a Skolem map over actions, so $\widetilde{\theta'}$ is as well. Then, consider the Skolem map $\theta'$ for which the function $\widetilde{\theta'}$ is its adjoint. By induction on the nesting of principal subsentences in $\varphi = \wp \flat \psi$, we now prove that $\mathcal{T} \models_{\theta'} \wp \flat \psi$. As base case, *i.e.*, when $\psi$ is an LTL formula, consider an assignment $\chi' \in \mathrm{Asg}_{\mathcal{T}}([\![\wp]\!], \varepsilon)$ and the induced play $\pi' = \mathsf{play}(\theta'(\chi') \circ \flat, \varepsilon)$ over $\mathcal{T}$. Moreover, consider an assignment $\chi \in \mathrm{Asg}_{\mathcal{G}}([\![\wp]\!], s_{\mathrm{o}})$ such that, for all placeholders $l \in \mathsf{dom}(\chi)$ and tracks $\rho' \in \mathrm{Trk}_{\mathcal{T}}$, it holds that $\chi(l)(\Gamma(\rho')) = \chi'(l)(\rho')$. From the satisfiability of $\varphi$ on $\mathcal{G}$, we derive that $\pi \models \psi$, where $\pi = \mathsf{play}(\theta(\chi) \circ \flat, s_{\mathrm{o}})$. Indeed, assume for a while that $(\pi)_{\leq k} = \Gamma((\pi)'_{\leq k})$. Then, from the definition of the labeling in the decision-tree unwinding, it easily follows that $\mathsf{ap}((\pi)_k) = \mathsf{ap}'((\pi')_k)$, and, so we derive $\pi' \models \psi$, from $\pi \models \psi$. Consequently, this holds for all $\chi' \in \mathrm{Asg}_{\mathrm{Str}_{\mathcal{T}}(\varepsilon)}([\![\flat]\!])$ and, so, we have that $\mathcal{T} \models_{\theta'} \wp \flat \psi$. The inductive case, *i.e.*, when $\psi$ contains some subsentence, easily follows by considering the principal subsentences as fresh atomic propositions.

It remains to prove, by induction on $k$, that $(\pi)_{\leq k} = \Gamma((\pi')_{\leq k})$, for all $k \in \mathbb{N}$. As base case, we have that $(\pi)_0 = s_{\mathrm{o}} = \Gamma(\varepsilon) = (\pi')_0$. As inductive case, assume that $(\pi)_{\leq k} = \Gamma((\pi')_{\leq k})$. Then, in particular, we have that $(\pi)_k = \Gamma((\pi')_k) = \mathsf{unw}((\pi')_k)$. By definition of play, it holds that $(\pi)_{k+1} = \mathsf{tr}((\pi)_k, (\widetilde{\theta}((\pi)_{\leq k}))(\widehat{\chi})) \circ \flat)$, which is, by inductive hypothesis, equal to $\mathsf{tr}(\mathsf{unw}((\pi')_k), (\widetilde{\theta}(\Gamma((\pi)_{\leq k})))(\widehat{\chi})) \circ \flat)$. Now, by the definition of $\widetilde{\theta'}$ and $\chi$, we obtain $\mathsf{tr}(\mathsf{unw}((\pi')_k), (\widetilde{\theta}(\Gamma((\pi)_{\leq k})))(\widehat{\chi})) \circ \flat) = \mathsf{tr}(\Gamma((\pi')_k), (\widetilde{\theta'}((\pi')_{\leq k}))(\widehat{\chi'})) \circ \flat)$. Finally, by the definition of $\mathsf{unw}$ and $\Gamma$, we have that $\mathsf{tr}(\Gamma((\pi')_k), (\widetilde{\theta'}((\pi')_{\leq k}))(\widehat{\chi'})) \circ \flat) = \Gamma(\pi')_{k+1}$. □

### 2.4.2 Bounded model property

In order to prove the bounded-tree model property for $\mathrm{SL}[1\mathrm{G}]$, we first need to introduce the new concept of *disjoint satisfiability*, which regards the verification of different instances of the same subsentence of the original specification. Intuitively, it asserts that either these instances can be checked on disjoint subtrees of the tree model or, if two instances use part of the same subtree, they are forced to use the same dependence map as well. This notion is a reformulation of the notion of explicit model introduced for ATL$^\star$ in [Sch08]. This intrinsic characteristic of $\mathrm{SL}[1\mathrm{G}]$ is fundamental for the building of a unique automaton that checks the truth of all subsentences, by simply merging their respective automata, without using a projection operation to eliminate their own alphabets, which otherwise may be in conflict. In this way, we are also able to avoid an exponential blow-up. A deeper discussion on this point is reported later in the paper.

**Definition 2.4.3 (Disjoint Satisfiability)** *Let $\mathcal{T}$ be a* DT *and $\varphi = \wp \flat \psi$ be a* $\mathrm{SL}[1\mathrm{G}]$ *principal sentence. Moreover, let* $\mathrm{S} \triangleq \{s \in \mathrm{St}_{\mathcal{T}} : \mathcal{T}, s \models \varphi\}$. *Then, $\mathcal{T}$ satisfies $\varphi$* disjointly *over* S *if there exist two functions* $\mathsf{head} : \mathrm{S} \to \mathrm{SM}_{\mathrm{Ac}}(\wp)$ *and* $\mathsf{body} : \mathrm{Trk}(\varepsilon) \to \mathrm{SM}_{\mathrm{Ac}}(\wp)$ *such that, for all $s \in \mathrm{S}$ and $\chi \in \mathrm{Asg}_{\mathrm{Str}(s)}([\![\wp]\!])$ it holds that $\mathcal{T}, \theta(\chi), s \models \flat\psi$, where the behavioral* $\mathtt{Sdf}$ $\theta \in \mathrm{BSDF}_{\mathrm{Str}(s)}$ *is defined, by means of its adjoint, as follows:*

*(i)* $\widetilde{\theta}(s) \triangleq \mathsf{head}(s)$*;*

*(ii)* $\widetilde{\theta}(\rho) \triangleq \mathsf{body}(\rho' \cdot \rho)$*, for all $\rho \in \mathrm{Trk}(s)$ with $|\rho| > 1$, where $\rho' \in \mathrm{Trk}(\varepsilon)$ is the unique track such that $\rho' \cdot \rho \in \mathrm{Trk}(\varepsilon)$* [5].

The disjoint satisfiability holds for all $\mathrm{SL}[1\mathrm{G}]$ formulas. To prove this fact, we first introduce the preliminary definition of *twin decision-tree*. Intuitively, in such a kind of tree, each action is flanked by a twin one having the same purpose of the original. This allows to satisfy two sentences requiring the same actions in a given state in two different branches of the tree itself, which is what the disjoint satisfiability precisely requires.

**Definition 2.4.4 (Twin Decision Tree)** *Let $\mathcal{T} = \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}, \mathrm{St}, \mathsf{tr}, \mathsf{ap}, \varepsilon \rangle$ be a* DT. *Then, the* twin decision tree *of $\mathcal{T}$ is the* DT $\mathcal{T}' \triangleq \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}', \mathrm{St}', \mathsf{tr}', \mathsf{ap}', \varepsilon' \rangle$ *with* $\mathrm{Ac}' = \mathrm{Ac} \times \{new, cont\}$ *and* $\varepsilon' = (\varepsilon, new)$. *The labeling and the transition function are defined by means of a set of projection functions introduced below:*

- *the function* $\mathsf{prj}_{\mathrm{Ac}} : \mathrm{Ac}' \to \mathrm{Ac}$ *returns the first component of the action in* $\mathrm{Ac}'$*,* i.e.*,* $\mathsf{prj}((c, \iota)) = c$*, for all $(c, \iota) \in \mathrm{Ac}'$;*
- *the function* $\mathsf{prj}_{\mathrm{Dc}} : \mathrm{Dc}' \to \mathrm{Dc}$ *projects out the flags on all the actions in the decision, returning a corresponding decision in $\mathcal{T}$,* i.e.*,* $\mathsf{prj}_{\mathrm{Dc}}(\delta')(a) = \mathsf{prj}_{\mathrm{Ac}}(\delta'(a))$*, for all $\delta' \in \mathrm{Dc}'$ and $a \in \mathrm{Ag}$;*
- *the function* $\mathsf{prj}_{\mathrm{St}} : \mathrm{St}' \to \mathrm{St}$ *returns the corresponding state in $\mathcal{T}$, according to the projection made on the decisions,* i.e.*,* $\mathsf{prj}_{\mathrm{St}}(\varepsilon') = \varepsilon$ *and* $\mathsf{prj}_{\mathrm{St}}(s' \cdot \delta') = \mathsf{prj}_{\mathrm{St}}(s') \cdot \mathsf{prj}_{\mathrm{Dc}}(\delta')$*, for all $s' \in \mathrm{St}'$ and $\delta' \in \mathrm{Dc}'$;*

[5] Existence and uniqueness of $\rho'$ is guaranteed by the fact that $\mathcal{T}$ is a DT.

- *analogously, the function* $\mathsf{prj}_{\mathrm{Trk}} : \mathrm{Trk}' \to \mathrm{Trk}$*, returns the concatenation of the projected states,* i.e.*,* $\mathsf{prj}_{\mathrm{Trk}}(s') = \mathsf{prj}_{\mathrm{St}}(s')$*, for all* $s' \in \mathrm{St}'$*, and* $\mathsf{prj}_{\mathrm{Trk}}(\rho' \cdot s') = \mathsf{prj}_{\mathrm{Trk}}(\rho') \cdot \mathsf{prj}_{\mathrm{St}}(s')$*, for all* $\rho' \in \mathrm{Trk}'$ *and* $s' \in \mathrm{St}'$*.*

*Then,* $\mathsf{ap}'(s') \triangleq \mathsf{ap}(\mathsf{prj}_{\mathrm{St}}(s'))$.

Observe that $\mathsf{prj}_{\mathrm{St}}(\mathsf{tr}'(s', \delta')) = \mathsf{tr}(\mathsf{prj}_{\mathrm{St}}(s'), \mathsf{prj}_{\mathrm{Dc}}(\delta'))$, for all $s' \in \mathrm{St}'$ and $\delta' \in \mathrm{Dc}'$. We can now prove the disjoint satisfiability property for $\mathrm{SL}[1\mathrm{G}]$.

**Theorem 2.4.2 (Disjoint Satisfiability)** *Let* $\varphi = \wp \flat \psi$ *be an* $\mathrm{SL}[1\mathrm{G}]$ *principal sentence and* $\mathcal{T} = \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}, \mathrm{St}, \mathsf{tr}, \mathsf{ap}, \varepsilon \rangle$ *a* DT*. Moreover, let* $\mathrm{S} \triangleq \{s \in \mathrm{St} : \mathcal{T}, s \models \varphi\}$*. Then the twin decision tree* $\mathcal{T}'$ *of* $\mathcal{T}$ *disjointly satisfies* $\varphi$ *over* $\mathrm{S}' \triangleq \{s' \in \mathrm{St}' : \mathsf{prj}(s') \in \mathrm{S}\}$.

*Proof.*[Proof idea] Starting from the fact that $\mathcal{T}, s \models \varphi$, for all $s \in \mathrm{S}$, by means of Theorem 2.2.4 on page 40, we derive the existence of a behavioral $\mathtt{Sdf}$ $\theta_s$. Such a $\theta_s$ is used to define a behavioral $\mathtt{Sdf}$ ${\theta_s}'$ in $\mathcal{T}'$ in which the existential agents suitably select either $new$ or $cont$ as second component, in order to guarantee the satisfaction of different instances over different branches of the twin decision tree. Indeed, it allows to properly define the two functions $\mathsf{head}$ and $\mathsf{body}$ and, consequently, the behavioral $\mathtt{Sdf}$ $\theta'$ for which we finally prove that $\mathcal{T}', s' \models \varphi$, for all $s' \in \mathrm{S}'$. Since $\theta'$ has been built from the head and body functions, the disjoint satisfiability is immediately derived. □

*Proof.* Let $s \in \mathrm{S}$ be one of the states on which $\varphi$ is satisfied. Since $\mathcal{T}, s \models \varphi$, by Theorem 2.2.4 on page 40, we have that there exists $\theta_s \in \mathrm{BSDF}_{\mathrm{Str}(s)}(\wp)$ such that $\mathcal{T}, \theta_s(\chi), s \models \flat\psi$, for all states $s$-total assignments $\chi \in \mathrm{Asg}_{\mathrm{Str}_{\mathcal{T}}(s)}([\![\wp]\!])$. Then, consider the adjoint function ${\widetilde{\theta_s}}' : \mathrm{Trk}' \rightharpoonup \mathrm{SM}_{\mathrm{Ac}'}(\wp)$ defined, for all states $s' \in \mathrm{St}'$, decisions $\delta' \in \mathrm{Dc}'$, and tracks $\rho' \in \mathrm{Trk}'$ as follows:

- ${\widetilde{\theta_s}}'(s')(\delta')(x) = (\widetilde{\theta_s}(\mathsf{prj}_{\mathrm{St}}(s'))(\mathsf{prj}_{\mathrm{Dc}}(\delta'))(x), new)$, if $\mathsf{prj}_{\mathrm{St}}(s') = s$;
- ${\widetilde{\theta_s}}'(\rho')(\delta')(x) = (\widetilde{\theta_s}(\mathsf{prj}_{\mathrm{Trk}}(\rho'))(\mathsf{prj}_{\mathrm{Dc}}(\delta'))(x), cont)$, otherwise.

At this point, we assume the function $\mathsf{head} : \mathrm{St}' \to \mathrm{SM}_{\mathrm{Ac}'}(\wp)$ to be defined as follows: $\mathsf{head}(s') \triangleq {\widetilde{\theta_s}}'(s')$. Moreover, we set the function $\mathsf{body} : \mathrm{Trk}'(\varepsilon) \to \mathrm{SM}_{\mathrm{Ac}'}(\wp)$ in such a way that it agrees with ${\widetilde{\theta_s}}'$ on all tracks $\rho' = {s_0}' \cdot \ldots \cdot {s_n}'$ for which there is an index $i \in \{0, \ldots, n\}$ such that, for all agents $a \in \mathrm{Ag}$ with $\flat(a) \in \langle\!\langle \wp \rangle\!\rangle$, it holds that:

- $\mathsf{lst}((\rho')_i)(a) = (c_a, new)$, for some $c_a \in \mathrm{Ac}$, and
- $\mathsf{lst}((\rho')_j)(a) = (c_a, cont)$, for all $j \in \{i+1, \ldots, n\}$ and for some $c_a \in \mathrm{Ac}$.

Note that the tracks of this form are such that the players bound to an existentially quantified variable have selected an action flagged by $new$ on the $i$-th step of the game and then keep playing with the $cont$ flag. Intuitively, they are starting the verification of a subsentence right in the $i$-th state of the track, by keeping it separated from the verification of the other subsentences, which are addressed with the $cont$ flag.

For all the other tracks $\rho'$, instead, the value of $\mathsf{body}(\rho')$ may be arbitrary.

Now, consider the behavioral $\mathtt{Sdf}\ \theta' \in \mathrm{BSDF}_{\mathrm{Str}}(\wp)$ defined by means of the functions $\mathsf{head}$ and $\mathsf{body}$ as prescribed by Definition 2.4.3 on page 53. It remains to prove that $\mathcal{T}', s' \models_{\theta'} \wp\flat\psi$, for all $s' \in \mathrm{S}'$. We proceed by induction on the nesting of the principal subsentences of $\varphi$. As base case, assume that such nesting is 0. This means that $\psi$ is an LTL formula. Now, let $\chi' \in \mathrm{Asg}_{\mathrm{Str}'(s')}([\![\wp]\!])$. By construction, it is not hard to see that there exists an assignment $\chi \in \mathrm{Asg}_{\mathrm{Str}(\mathsf{prj}(s'))}([\![\wp]\!])$ for which the play $\pi' \triangleq \mathsf{play}'(\theta'(\chi') \circ \flat, s')$ satisfies the equality $\mathsf{prj}_{\mathrm{Pth}}(\pi') = \mathsf{play}(\theta(\chi) \circ \flat, \mathsf{prj}(s')) = \pi$ [6]. Thus, since $\mathcal{T}, s \models_{\theta} \wp\flat\psi$, we have that $\pi \models \psi$. Moreover, it holds that $\mathsf{ap}'((\pi')_i) = \mathsf{ap}((\pi)_i)$, for all $i \in \mathbb{N}$, which implies that $\pi' \models \psi$. Consequently, we can conclude that $\mathcal{T}', s' \models_{\theta'} \wp\flat\psi$. The inductive case, easily follows by considering the inner principal subsentences as fresh atomic propositions. □

We now have all tools to prove the bounded model property of $\mathrm{SL}[1\mathrm{G}]$.

**Theorem 2.4.3 (Bounded Model Property of $\mathrm{SL}[1\mathrm{G}]$)** *Let $\varphi$ be a $\mathrm{SL}[1\mathrm{G}]$ sentence and $\mathcal{T}$ be a* DT *such that $\mathcal{T} \models \varphi$. Then, there exists a bounded* DT *$\mathcal{T}'$ such that $\mathcal{T}' \models \varphi$.*

The proof makes use of some instruments and formalisms for *First-Order Logic* (FOL, for short) that are introduced in [MP14]. For the sake of completeness, here we give an informal discussion of such object. A *language signature* is a tuple $\mathcal{L} = \langle \mathrm{Ar}, \mathrm{Rl}, \mathsf{ar} \rangle$ in which $\mathrm{Ar}$ and $\mathrm{Rl}$ are two finite non-empty sets of *arguments* and *relations*, respectively, and $\mathsf{ar} : \mathrm{Rl} \to 2^{\mathrm{Ar}} \setminus \{\emptyset\}$ is a function mapping each relation in $\mathrm{Rl}$ to its non-empty set of arguments. Language signatures are used to reformulate FOL syntax in terms of *binding forms*, which are a way to associate variables to relations by means of bindings. The interpretation of FOL formulas is given on *relational structures*, which are tuples $\mathcal{R} = \langle \mathrm{Dm}, \mathsf{rl} \rangle$ with $\mathrm{Dm}$ being a non-empty domain and where $\mathsf{rl}(r) \subseteq \mathsf{ar}(r) \to \mathrm{Dm}$ is a set of functions, representing the tuples on which the relation $r \in \mathrm{Rl}$ is interpreted as true.

*Proof.*[Proof Idea] The key idea used to prove the theorem is based on the *finite model property* of the *One-Binding* fragment of FOL (FOL[1B], for short), proved in [MP14], which allows to define a bounded-tree model $\mathcal{T}'$, which preserves the satisfiability of $\varphi$. In particular, for each state $s^*$ of a tree $\mathcal{T}$ satisfying $\varphi$, we build a first-order structure and a FOL[1B] formula $\eta_{s^*}$ that characterizes the topology of the successors of $s_*$ in $\mathcal{T}$. Then, since FOL[1B] enjoys the finite model property, we are able to build a finite first-order structure for $\eta_{s^*}$ from which we can build the bounded model $\mathcal{T}'$ of $\varphi$. Such a construction is based on both the disjoint and behavioral satisfiability of $\mathrm{SL}[1\mathrm{G}]$. For each state $s^*$ in $\mathcal{T}$, we consider a set given by pairs of subsentences $\eta$ of $\varphi$ and states $s$, on which it holds that $\mathcal{T}, s \models_{\theta} \eta$, where such satisfaction is forced to pass through $s^*$ for at least one universal assignment fed to $\theta$. This means that at least one play used to satisfy $\eta$ passes through $s^*$. By the disjoint satisfiability, we have to cope with at most two $\mathtt{Sdf}$s for each subsentence $\eta$, those given by $\mathsf{head}_\eta$ and $\mathsf{body}_\eta$, implying that the total number of $\mathtt{Sdf}$s to take into account, for all $s^*$, is finite. From that, we define a related FOL[1B] sentence $\eta_{s^*}$, having a model derived from the topology of the successors of $s^*$ whose elements are constituted by the actions of $\mathcal{T}$. Now, by applying the finite model property to $\eta_{s^*}$, we derive the existence of a model for the formula $\eta_{s^*}$ with a finite domain $\mathrm{Ac}'_{\eta,s^*}$. Exploiting the finite model built for all states $s^*$, we are able to define the labeling of $\mathcal{T}'$ and a behavioral $\mathtt{Sdf}\ \theta'$ in such a way that $\mathcal{T}', \varepsilon \models_{\theta'} \varphi$. □

[6] By $\mathsf{prj}_{\mathrm{Pth}}$ we are denoting the natural lifting of the function $\mathsf{prj}_{\mathrm{Trk}}$ to paths.

*Proof.*[Proof of Theorem 2.4.3 on the preceding page] We give the proof for the case of $\varphi = \wp\flat\psi$, since the Boolean combination of principal sentences easily follows from this one. Given a tree-model $\mathcal{T}$ for $\varphi$, derived by the tree-model property of SL[1G] of Theorem 2.4.1 on page 52, for each state $s^* \in \mathrm{St}$, consider the set $\Phi_{s^*} \subseteq \mathrm{St} \times \mathsf{psnt}(\varphi)$ of states of $\mathcal{T}$ and principal subsentences of $\varphi$ such that $(s, \eta) \in \Phi_{s^*}$ iff *(i)* $\mathcal{T}, s \models \eta$ and *(ii)* there exists an assignment $\chi \in \mathrm{Asg}_{\mathrm{Str}(s)}([\![\wp\eta]\!])$ such that $s^* = (\mathsf{play}(\theta^{s^*}_{(s,\eta)})(\chi)) \circ \flat_\eta, s)_n$, for some $n \in \mathbb{N}$, where the behavioral $\mathtt{Sdf}$ $\theta^{s^*}_{(s,\eta)}$ is defined by means of its adjoint, which is in its turn built from the functions $\mathsf{head}_\eta$ and $\mathsf{body}_\eta$, given by Theorem 2.4.2 on page 54, applied on $\eta = \wp\flat\psi_\eta$. Observe that, for a fixed $\eta$, if $s_1, s_2 \in \mathrm{St}$, with $s_1 \neq s^*$ and $s_2 \neq s^*$, $\widetilde{\theta^{s^*}_{(s,\eta)}}(\rho_{s_1}) = \mathsf{head}_\eta(\rho^{'}_{s_1} \cdot \rho_{s_1}) = \mathsf{head}_\eta(\rho^{'}_{s_2} \cdot \rho_{s_2}) = \widetilde{\theta^{s^*}_{(s,\eta)}}(\rho_{s_2})$, where $\rho_{s_1}$ and $\rho_{s_2}$ are the unique tracks ending in $s^*$ and starting in $s_1$ and $s_2$, respectively, while $\rho^{'}_{s_1}$ and $\rho^{'}_{s_2}$ are the unique tracks such that $\rho^{'}_{s_1} \cdot \rho_{s_1}$ and $\rho^{'}_{s_2} \cdot \rho_{s_2}$ start from $\varepsilon$. Now, for a given $\mathtt{Sdf}$ over Actions $\widetilde{\theta} \in \mathrm{SM}_{\mathrm{Ac}}(\wp)$ and a given state $s \in \mathrm{St}$, define the set $Succ_{\vartheta,\flat}(s) \triangleq \{s' \in \mathrm{St} : \exists \mathsf{v} \in \mathrm{Ac}^{[\![\wp]\!]}.\mathsf{tr}(s, \vartheta(\mathsf{v}) \circ \flat) = s'\}$, where $\vartheta \in \mathrm{Ac}^{[\![\wp]\!]} \to \mathrm{Ac}^{\mathrm{Vr}(\wp)}$ is a $\mathtt{Sdf}$ for $\wp$ over actions. Intuitively, the set $Succ_{\vartheta,\flat}(s)$ defines the set of states that can be reached in one step from $s$ by prescribing the agents that are bound by $\flat$ to an existential variable to move according to the $\mathtt{Sdf}$ $\vartheta$.

At this point, consider the language signature $\mathcal{L} = \langle \mathrm{Ar}, \mathrm{Rl}, \mathsf{ar}\rangle = \langle \mathrm{Ag}, \mathrm{AP}, \mathsf{ar}\rangle$ with $\mathsf{ar}(p) = \mathrm{Ag}$, for all $p \in \mathrm{AP}$, where each atomic proposition is viewed as a relation having the agents as arguments and, so, the decisions as elements of its interpretation. Moreover, for all sets $\mathrm{P} \subseteq \mathrm{AP}$, let $\mathsf{mask}^{\mathrm{P}} = \bigwedge_{p\in \mathrm{P}} p \wedge \bigwedge_{q\in \mathrm{AP}\setminus \mathrm{P}} \neg q$ be the FOL[1B] formula asserting that only the relations in P hold. Finally, for all $(s, \eta) \in \Phi_{s^*}$, consider the FOL[1B] sentence $\eta^*_s = \wp\eta\flat_\eta \bigvee_{s\in Succ_{\theta^{s^*}_{(s,\eta)})}(trk(s^*),\flat_\eta}} \mathsf{mask}^{\mathsf{ap}(s)}$ [7]. Clearly, by definition, each $\eta^*_s$ is satisfied by the relational structure $\mathcal{R}_{s^*} = \langle \mathrm{Ac}, \mathsf{rl}_{s^*}\rangle$ with $\mathsf{rl}_{s^*}(p) \triangleq \{\delta \in \mathrm{Dc} : p \in \mathsf{ap}(\mathsf{tr}(s^*, \delta)\}$, where a relation $p$ is interpreted as true on all decisions that allow $s^*$ to reach a state in which $p$ holds. Indeed, for each partial valuation $\mathsf{v} \in \mathrm{Ac}^{[\![\wp\eta]\!]}$, it holds that either $s' = \mathsf{tr}(s, \mathsf{head}_\eta(s)(\mathsf{v})) \in Succ_{\mathsf{head}_\eta(s),\flat_\eta}(s)$ or $s'' = \mathsf{tr}(s, \mathsf{body}_\eta(trk(s))(\mathsf{v})) \in Succ_{\mathsf{body}_\eta(trk(s)),\flat_\eta}(s)$, which implies that either $\mathcal{R}_{s^*}, \mathsf{head}_\eta(s)(\mathsf{v}) \models mask^{\mathsf{ap}(s')}$ or $\mathcal{R}_{s^*}, \mathsf{body}_\eta(trk(s))(\mathsf{v}) \models mask^{\mathsf{ap}(s'')}$. Hence, we have that $\mathcal{R}_{s^*} \models \eta^*_s$ and, since this is true for all $(s, \eta) \in \Phi_{s^*}$, we derive that $\mathcal{R}_{s^*} \models \bigwedge_{(s,\eta)\in\Phi_{s^*}} \eta^*_s$.

At this point, from the finite model property of FOL[1B], we derive that there exists a finite relational structure $\mathcal{R}^{'}_{s^*} = \langle \mathrm{Dm}^{'}_{s^*}, \mathsf{rl}^{'}_{s^*}\rangle$ such that $\mathcal{R}^{'}_{s^*} \models \bigwedge_{(s,\eta)\in\Phi_{s^*}} \eta^*_s$. Moreover, we define $\widetilde{\theta'^{s^*}_{(s,\eta)}}$ to be such that $\mathcal{R}^{'}_{s^*} \models_{\widetilde{\theta'^{s^*}_{(s,\eta)}}} \eta^*_s$. Observe that, in the proof of finite model property for FOL[1B] [MP14], the bound on $\mathrm{Dm}^{'}_{s^*}$ only depends on the quantification and binding prefixes given in the formulas $\eta^*_s$, which all occur in $\varphi$. Thus, the size of $\mathrm{Dm}^{'}_{s^*}$ does not depend on $\eta$ and, *w.l.o.g.*, we can assume that $\mathrm{Dm}^{'}_{s^*} = \mathrm{Ac}'$ for all $s^* \in \mathrm{St}$. Moreover, again from the finite model property proof of FOL[1B], there exists a function $\mathsf{M}_{s^*} : \mathrm{Dc}' \to \mathrm{Dc}$, with $\mathrm{Dc}' = \mathrm{Ac}'^{\mathrm{Ag}}$, such that, for all $(s, \eta) \in \Phi_{s^*}$ and $\delta' \in \mathsf{rng}(\widetilde{\theta'^{s^*}_{(s,\eta)}})$, we have that $\mathsf{M}_{s^*}(\delta') \in \mathsf{rng}(\widetilde{\theta^{s^*}_{(s,\eta)}})$, where $\widetilde{\theta^{s^*}_{(s,\eta)}}$ is the $\mathtt{Sdf}$ used to satisfy $\eta^*_s$ on $\mathcal{R}$. At this point, we define a DT $\mathcal{T}'$ having $\mathrm{Ac}'$ as set of

[7] By $trk(s)$ we are denoting the unique track starting from $\varepsilon$ and ending in $s$.

actions. In order to define the labeling function $\mathsf{ap}'$, first consider the mapping $\Gamma : \mathrm{St}' \to \mathrm{St}$ recursively defined as follows:

- $\Gamma(\varepsilon) = \varepsilon$;
- $\Gamma(s' \cdot \delta') = \Gamma(s') \cdot \mathsf{M}_{\Gamma(s')}(\delta')$.

By means of $\Gamma$, define $\mathsf{ap}'(s') \triangleq \mathsf{ap}(\Gamma(s'))$ for all $s' \in \mathrm{St}'$. It remains to prove that $\mathcal{T}' \models \varphi$. We do this by using a $\mathtt{Sdf}\ \theta' \in \mathrm{SM}_{\mathrm{Str}'(\varepsilon)}(\wp)$ defined from the adjoint $\widetilde{\theta'}$ introduced in the following.

Let $\rho'$ be a track in $\mathcal{T}'$ and consider $s' = \mathsf{lst}(\rho')$. If $(\varepsilon, \varphi) \in \Phi_{\Gamma(s')}$, define $\widetilde{\theta'}(\rho') = \theta'^{s^*}_{(s,\eta)}$. For all other tracks, the value of $\widetilde{\theta'}$ may be arbitrary. We now show that $\mathcal{T}, \varnothing, \varepsilon \models_{\theta'} \varphi$, by induction on the nesting of principal subsentences. As base case, suppose that $\varphi$ has nesting 0. This implies that it is of the form $\wp \flat \psi$ with $\psi$ being an LTL formula. Then, consider a universal assignment $\chi' \in \mathrm{Asg}_{\mathrm{Str}'(\varepsilon)}([\![\wp]\!])$ and then the total assignment $\theta'(\chi')$. This determines a play $\pi' = \mathsf{play}(\theta'(\chi'), \varepsilon)$. Now, consider a universal assignment $\chi \in \mathrm{Asg}_{\mathrm{Str}(\varepsilon)}([\![\wp]\!])$ such that, for all $x \in [\![\wp]\!]$ and $\rho' \in \mathrm{Trk}'(\varepsilon)$, it holds that $\chi'(x)(\rho') = \chi(x)(\Gamma(\rho'))$, where $\Gamma$ is the lifting over tracks of the mapping over states defined above, *i.e.*, by $\Gamma(\rho')$ is the track in $\mathcal{T}$ obtained from $\rho'$ by mapping each state $s'$ in $\rho'$ into $s = \Gamma(s')$. It holds that $\pi = \mathsf{play}(\theta(\chi) \circ \flat, \varepsilon)$ is such that, for all $i \in \mathbb{N}$, $(\pi)_{\leq i} = \Gamma((\pi')_{\leq i})$. Indeed, by induction on $i$, as base case, we have that $(\pi)_{\leq 0} = \varepsilon = \Gamma(\varepsilon) = (\pi')_{\leq 0}$. As inductive case, suppose that $(\pi)_{\leq i} = \Gamma((\pi')_{\leq i})$. Then, $(\pi)_{i+1} = \mathsf{tr}((\pi)_i, \widetilde{\theta}((\pi)_{\leq i}) \circ \flat) = \mathsf{tr}(\Gamma((\pi')_i), \widetilde{\theta}(\Gamma((\pi')_{\leq i})) \circ \flat) = \Gamma(\mathsf{tr}'((\pi')_i, \widetilde{\theta'}((\pi')_{\leq i}) \circ \flat)) = \Gamma((\pi')_{i+1})$. Thus, according to the definition of $\mathsf{ap}'$, we have that $\mathsf{ap}'((\pi')_i) = \mathsf{ap}((\pi)_i)$, for all $i \in \mathbb{N}$. Since $\pi \models \psi$, we derive that $\pi' \models \psi$. This holds for all possible universal assignments $\chi' \in \mathrm{Asg}_{\mathrm{Str}'(\varepsilon)([\![\wp]\!])}$. Hence, it holds that $\mathcal{T}' \models_{\theta'} \wp \flat \psi$. The inductive case follows by considering the principal subsentences of $\varphi$ as fresh atomic propositions. □

### 2.4.3 Satisfiability procedure

We finally solve the satisfiability problem for SL[1G] and show that it is 2EXPTIME-COMPLETE, as for ATL$^\star$. 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) [GTW02]. From an high-level point of view, the automaton construction seems similar to what was proposed the in literature for CTL$^\star$ [KVW00] and ATL$^\star$ [Sch08]. However, our technique is completely new, since it is based on the novel notions of behavioral semantics and disjoint satisfiability.

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

**Definition 2.4.5 (Assignment-Labeling Encoding)** *Let $\mathcal{T}$ be a* DT*, $t \in \mathrm{St}_\mathcal{T}$ one of its states, and $\chi \in \mathrm{Asg}_\mathcal{T}(\mathrm{V}, t)$ a $t$-total assignment defined on the set $\mathrm{V} \subseteq \mathrm{Vr}$. 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 $\chi$ on $\mathcal{T}$ if $\mathsf{u}(\mathsf{lst}((\rho)_{\geq 1})) = (\widehat{\chi}(\rho), \mathsf{ap}_\mathcal{T}(\mathsf{lst}(\rho)))$, for all $\rho \in \mathrm{Trk}_\mathcal{T}(t)$* [8].

[8] Note that $\mathsf{lst}(\varepsilon) = \varepsilon$.

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. The $\mathcal{U}^{\mathrm{Ac}}_{\flat\psi}$ recognizes all the assignment-labeling encodings $\mathcal{T}'$ of an a priori given assignment $\chi$ over a generic DT $\mathcal{T}$, whenever the goal is satisfied on $\mathcal{T}$ under $\chi$. Intuitively, we start with a UCW, recognizing all infinite words on the alphabet $2^{\mathrm{AP}}$ that satisfy the LTL formula $\psi$, obtained by a simple variation of the Vardi-Wolper construction [VW86]. Then, we run it on the encoding tree $\mathcal{T}'$ by following the directions identified by the assignment in its labeling.

**Lemma 2.4.1 (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 DT $\mathcal{T}$ with $\mathrm{Ac}_{\mathcal{T}} = \mathrm{Ac}$, states $t \in \mathrm{St}_{\mathcal{T}}$, and $t$-total 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 $\chi$ on $\mathcal{T}$.*

*Proof.* A first step in the construction of the UCT $\mathcal{U}^{\mathrm{Ac}}_{\flat\psi}$, is to consider the UCW $\mathcal{U}_\psi \triangleq \langle 2^{\mathrm{AP}}, \mathrm{Q}_\psi, \delta_\psi, \mathrm{Q}_{\mathrm{o}\psi}, \aleph_\psi\rangle$ obtained by dualizing the NBW resulting from the application of the classic Vardi-Wolper construction to the LTL formula $\neg\psi$ [VW86]. Observe that $\mathrm{L}(\mathcal{U}_\psi) = \mathrm{L}(\psi)$, *i.e.*, this automaton recognizes all infinite words on the alphabet $2^{\mathrm{AP}}$ that satisfy the LTL formula $\psi$. Then, define the components of $\mathcal{U}^{\mathrm{Ac}}_{\flat\psi} \triangleq \langle \mathrm{Val}_{\mathrm{Ac}}(\mathsf{free}(\flat\psi)) \times 2^{\mathrm{AP}}, \mathrm{Dc}, \mathrm{Q}_{\flat\psi}, \delta_{\flat\psi}, q_{\mathrm{o}\flat\psi}, \aleph_{\flat\psi}\rangle$, as follows:

- $\mathrm{Q}_{\flat\psi} \triangleq \{q_{\mathrm{o}\flat\psi}\} \cup \mathrm{Q}_\psi$, with $q_{\mathrm{o}\flat\psi} \notin \mathrm{Q}_\psi$;
- $\delta_{\flat\psi}(q_{\mathrm{o}\flat\psi}, (\mathsf{v}, \sigma)) \triangleq \bigwedge_{q \in \mathrm{Q}_{\mathrm{o}\psi}} \delta_{\flat\psi}(q, (\mathsf{v}, \sigma))$, for all $(\mathsf{v}, \sigma) \in \mathrm{Val}_{\mathrm{Ac}}(\mathsf{free}(\flat\psi)) \times 2^{\mathrm{AP}}$;
- $\delta_{\flat\psi}(q, (\mathsf{v}, \sigma)) \triangleq \bigwedge_{q' \in \delta_\psi(q,\sigma)} (\mathsf{v} \circ \flat, q')$, for all $q \in \mathrm{Q}_\psi$ and $(\mathsf{v}, \sigma) \in \mathrm{Val}_{\mathrm{Ac}}(\mathsf{free}(\flat\psi)) \times 2^{\mathrm{AP}}$;
- $\aleph_{\flat\psi} \triangleq \aleph_\psi$.

Intuitively, the UCT $\mathcal{U}^{\mathrm{Ac}}_{\flat\psi}$ simply runs the UCW $\mathcal{U}_\psi$ on the branch of the encoding individuated by the assignment in input. Thus, it is easy to see that, for all states $t \in \mathrm{St}_{\mathcal{T}}$ and $t$-total 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 $\chi$ on $\mathcal{T}$. □

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

**Definition 2.4.6 (Behavioral Dependence-Labeling Encoding)** *Let $\mathcal{T}$ be a DT, $t \in \mathrm{St}_{\mathcal{T}}$ one of its states, and $\theta \in \mathrm{BSDF}_{\mathrm{Str}_{\mathcal{T}}(t)}(\wp)$ a behavioral dependence map over strategies for a quantification prefix $\wp \in \mathrm{Qnt}(\mathrm{V})$ over the set $\mathrm{V} \subseteq \mathrm{Vr}$. A $(\mathrm{SM}_{\mathrm{Ac}_{\mathcal{T}}}(\wp) \times 2^{\mathrm{AP}})$-labeled Dir-tree $\mathcal{T}' \triangleq \langle \mathrm{St}_{\mathcal{T}}, \mathsf{u}\rangle$ is a* behavioral dependence-labeling encoding *for $\theta$ on $\mathcal{T}$ if $\mathsf{u}(\mathsf{lst}((\rho)_{\geq 1})) = (\widetilde{\theta}(\rho), \mathsf{ap}_{\mathcal{T}}(\mathsf{lst}(\rho)))$, for all $\rho \in \mathrm{Trk}_{\mathcal{T}}(t)$.*

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

Finally, in the next lemma, we show how to locally handle 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 2.4.2 (**$\mathrm{SL}[1\mathrm{G}]$ **Sentence Automaton)** *Let* $\wp\flat\psi$ *be an* $\mathrm{SL}[1\mathrm{G}]$ *principal sentence without principal subsentences and* $\mathrm{Ac}$ *a finite set of actions. Then, there exists an* $\textsc{Uct}$ $\mathcal{U}^{\mathrm{Ac}}_{\wp\flat\psi} \triangleq \langle \mathrm{SM}_{\mathrm{Ac}}(\wp) \times 2^{\mathrm{AP}}, \mathrm{Dc}, \mathrm{Q}_{\wp\flat\psi}, \delta_{\wp\flat\psi}, q_{\mathrm{o}\wp\flat\psi}, \aleph_{\wp\flat\psi}\rangle$ *such that, for all* DT *s* $\mathcal{T}$ *with* $\mathrm{Ac}_{\mathcal{T}} = \mathrm{Ac}$*, states* $t \in \mathrm{St}_{\mathcal{T}}$*, and behavioral dependence maps over strategies* $\theta \in \mathrm{BSDF}_{\mathrm{Str}_{\mathcal{T}}(t)}(\wp)$*, it holds that* $\mathcal{T}, \theta(\chi), t \models_{\mathrm{B}} \flat\psi$*, for all* $\chi \in \mathrm{Asg}_{\mathcal{T}}([\![\wp]\!], t)$*, iff* $\mathcal{T}' \in \mathrm{L}(\mathcal{U}^{\mathrm{Ac}}_{\flat\psi})$*, where* $\mathcal{T}'$ *is the behavioral dependence-labeling encoding for* $\theta$ *on* $\mathcal{T}$*.*

*Proof.* By Lemma 2.4.1 on the preceding page of $\mathrm{SL}[1\mathrm{G}]$ goal automaton, there is an $\textsc{Uct}$ $\mathcal{U}^{\mathrm{Ac}}_{\flat\psi} \triangleq \langle \mathrm{Val}_{\mathrm{Ac}}(\mathsf{free}(\flat\psi)) \times 2^{\mathrm{AP}}, \mathrm{Dc}, \mathrm{Q}_{\flat\psi}, \delta_{\flat\psi}, q_{0\flat\psi}, \aleph_{\flat\psi}\rangle$ 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 $\chi$ on $\mathcal{T}$.

Now, transform $\mathcal{U}^{\mathrm{Ac}}_{\flat\psi}$ into the new $\textsc{Uct}$ $\mathcal{U}^{\mathrm{Ac}}_{\flat\psi} \triangleq \langle \mathrm{SM}_{\mathrm{Ac}}(\wp) \times 2^{\mathrm{AP}}, \mathrm{Dc}, \mathrm{Q}_{\wp\flat\psi}, \delta_{\wp\flat\psi}, q_{0\wp\flat\psi}, \aleph_{\wp\flat\psi}\rangle$, with $\mathrm{Q}_{\wp\flat\psi} \triangleq \mathrm{Q}_{\flat\psi}$, $q_{\mathrm{o}\wp\flat\psi} \triangleq q_{\mathrm{o}\flat\psi}$, and $\aleph_{\wp\flat\psi} \triangleq \aleph_{\flat\psi}$, which is used to handle the quantification prefix $\wp$ atomically, where the transition function is defined as follows: $\delta_{\wp\flat\psi}(q, (\theta, \sigma)) \triangleq \bigwedge_{\mathsf{v} \in \mathrm{Val}_{\mathrm{Ac}}([\![\wp]\!])} \delta_{\flat\psi}(q, (\theta(\mathsf{v}), \sigma))$, for all $q \in \mathrm{Q}_{\wp\flat\psi}$ and $(\theta, \sigma) \in \mathrm{SM}_{\mathrm{Ac}}(\wp) \times 2^{\mathrm{AP}}$. Intuitively, $\mathcal{U}^{\mathrm{Ac}}_{\wp\flat\psi}$ reads an action dependence map $\theta$ on each node of the input tree $\mathcal{T}'$ labeled with a set of atomic propositions $\sigma$ and simulates the execution of the transition function $\delta_{\flat\psi}(q, (\mathsf{v}, \sigma))$ of $\mathcal{U}^{\mathrm{Ac}}_{\flat\psi}$, for each possible valuation $\mathsf{v} = \theta(\mathsf{v}')$ on $\mathsf{free}(\flat\psi)$ obtained from $\theta$ via a universal valuation $\mathsf{v}' \in \mathrm{Val}_{\mathrm{Ac}}([\![\wp]\!])$. It is worth observing that we cannot move the component set $\mathrm{SM}_{\mathrm{Ac}}(\wp)$ from the input alphabet to the states of $\mathcal{U}^{\mathrm{Ac}}_{\wp\flat\psi}$ by making a related guessing of the dependence map $\theta$ in the transition function, since the automaton is universal and we have to ensure that all states in a given node of the tree $\mathcal{T}'$, *i.e.*, in each track of the original model $\mathcal{T}$, make the same choice for $\theta$.

Finally, it remains to prove that, for all states $t \in \mathrm{St}_{\mathcal{T}}$ and behavioral dependence maps over strategies $\theta \in \mathrm{BSDF}_{\mathrm{Str}_{\mathcal{T}}(t)}(\wp)$, it holds that $\mathcal{T}, \theta(\chi), t \models_{\mathrm{B}} \flat\psi$, 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 behavioral dependence-labeling encoding for $\theta$ on $\mathcal{T}$.

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

*[If].* Suppose that $\mathcal{T}' \in \mathrm{L}(\mathcal{U}^{\mathrm{Ac}}_{\wp\flat\psi})$. Then, there exists a $(\mathrm{Dc} \times \mathrm{Q}_{\wp\flat\psi})$-tree R that is an accepting run for $\mathcal{U}^{\mathrm{Ac}}_{\wp\flat\psi}$ on $\mathcal{T}'$. Now, for each $\chi \in \mathrm{Asg}_{\mathcal{T}}([\![\wp]\!], t)$, let $\mathrm{R}_{\chi}$ be the run for $\mathcal{U}^{\mathrm{Ac}}_{\flat\psi}$ on the assignment-state encoding $\mathcal{T}'_{\chi}$ for $\theta(\chi)$ on $\mathcal{T}$. Due to the particular definition of the

transition function of $\mathcal{U}^{\mathrm{Ac}}_{\wp\flat\psi}$, it is not hard to see that $\mathrm{R}_\chi \subseteq \mathrm{R}$. Thus, since R is accepting, we have that $\mathrm{R}_\chi$ is accepting as well. So, $\mathcal{T}'_\chi \in \mathrm{L}(\mathcal{U}^{\mathrm{Ac}}_{\flat\psi})$. At this point, due to the property of $\mathcal{U}^{\mathrm{Ac}}_{\flat\psi}$, it follows that $\mathcal{T}, \theta(\chi), t \models \flat\psi$. Since $\psi$ does not contain principal subsentences, we have that $\mathcal{T}, \theta(\chi), t \models_{\mathrm{B}} \flat\psi$, for all $\chi \in \mathrm{Asg}_{\mathcal{T}}([\![\wp]\!], t)$. □

**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 $\varphi$, 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 of Definition 2.4.3 on page 53, 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 2.4.3 on page 55 on the bounded-tree model property.

**Theorem 2.4.4 (SL[1G] Automaton)** *Let $\varphi$ be an* SL[1G] *sentence. Then, there exists an* UCT $\mathcal{U}_\varphi$ *such that $\varphi$ 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 2.4.5 (SL[1G] Satisfiability)** *The satisfiability problem for* SL[1G] *is* 2EXPTIME-COMPLETE.

*Proof.* By Theorem 2.4.4 of SL[1G] automaton, to verify whether an SL[1G] sentence $\varphi$ is satisfiable we can calculate the emptiness of the UPT $\mathcal{U}_\varphi$. This automaton is obtained by merging all UCTs $\mathcal{U}^{\mathrm{Ac}}_{\phi}$, with $\phi = \wp\flat\psi \in \mathsf{psnt}(\varphi)$, which in turn are based on the UCTs $\mathcal{U}^{\mathrm{Ac}}_{\flat\psi}$ that embed the UCWs $\mathcal{U}_\psi$. By a simple calculation, it is easy to see that $\mathcal{U}_\varphi$ has $2^{\mathrm{O}(|\varphi|)}$ states.

Now, by using a well-known nondeterminization procedure for APTs [MS95], we obtain an equivalent NPT $\mathcal{N}_\varphi$ with $2^{2^{\mathrm{O}(|\varphi|)}}$ states and index $2^{\mathrm{O}(|\varphi|)}$.

The emptiness problem for such a kind of automaton with $n$ states and index $h$ is solvable in time $\mathrm{O}(n^h)$.Thus, we get that the time complexity of checking whether $\varphi$ is satisfiable is $2^{2^{\mathrm{O}(|\varphi|)}}$. Hence, the membership of the satisfiability problem for SL[1G] in 2EXPTIME directly follows. Finally the thesis is proved, by getting the relative lower bound from the same problem for CTL$^\star$ [VS85]. □

CHAPTER 3

# Synthesis

The chapter is organized as follows. In Section 3.1 we provide the basic notions and formal definitions of rational synthesis. Then, in Section 3.2 we introduce and solve the problem of *qualitative rational synthesis*, showing that it is in 2EXPTIME-COMPLETE. Finally, in Section 3.3 we address also a quantitative variant of rational synthesis, proving that it is in 2EXPTIME-COMPLETE, as well as the qualitative case.

## 3.1 Rational synthesis

We define two variants of rational synthesis. The first, *cooperative rational synthesis*, was introduced in [FKL10]. The second, *non-cooperative rational synthesis*, was introduced in [KPV14].

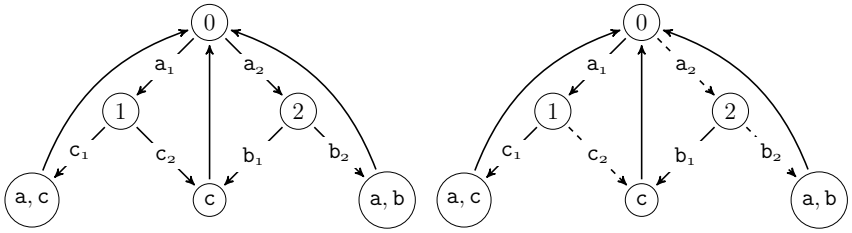
We work with the following model: the world consists of a *system* and an environment composed of $k$ agents: $\alpha_1, \ldots, \alpha_k$. For uniformity, we refer to the system as Agent $\alpha_0$. We assume that Agent $\alpha_i$ controls a set $\mathrm{X}_i$ of propositions, and the different sets are pairwise disjoint. At each point in time, each agent sets his propositions to certain values. Let $\mathrm{X} = \bigcup_{0 \leq i \leq k} \mathrm{X}_i$, and $\mathrm{X}_{-i} = \mathrm{X} \setminus \mathrm{X}_i$. Each agent $\alpha_i$ (including the system) has an objective $\varphi_i$, specified as an LTL formula over X.

This setting induces the CGS $\mathcal{G} = \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}, \mathrm{St}, \mathsf{tr}, \mathsf{ap}, s_0 \rangle$ defined as follows. The set of agents $\mathrm{Ag} = \{\alpha_0, \alpha_1, \ldots, \alpha_k\}$ consists of the system and the agents that constitute the environment. The actions of Agent $\alpha_i$ are the possible assignments to its variables. Thus, $\mathrm{Ac}_i = 2^{\mathrm{X}_i}$. We use Ac and $\mathrm{Ac}_{-i}$ to denote the sets $2^{\mathrm{X}}$ and $2^{\mathrm{X}_{-i}}$, respectively. The nodes of the game record the current assignment to the variables. Hence, $\mathrm{St} = \mathrm{Ac}$, and for all $s \in \mathrm{St}$ and $\langle c_0, \ldots, c_k \rangle \in \mathrm{Ac}_0 \times \mathrm{Ac}_1 \times \cdots \times \mathrm{Ac}_k$, we have $\delta(s, c_0, \ldots, c_k) = \langle c_0, \cdots, c_k \rangle$.

A strategy for the system is a function $\mathsf{f}_0 : \mathrm{Trk} \to \mathrm{Ac}_0$. In the standard synthesis problem, we say that $\mathsf{f}_0$ realizes $\varphi_0$ if, no matter which strategies the agents composing the environment follow, all the paths in which the system follows $\mathsf{f}_0$ satisfy $\varphi_0$. In rational synthesis, on instead, we assume that the agents that constitute the environment are rational, which soften the universal quantification on the behavior of the environment.

Recall that the rational-synthesis problem gets a solution concept as a parameter. As discussed in the introduction, the fact that a strategy profile is a solution with respect to the concept guarantees that it is not worthwhile for the agents constituting the environment to deviate from the strategies assigned to them. Several solution concepts are studied and motivated in game theory. Here, we focus on the concepts of *dominant strategy* and *Nash equilibrium*, defined below.

The common setting in game theory is that the objective for each agent is to maximize his *payoff* – a real number that is a function of the outcome of the game. We use $\mathsf{payoff}_i$ :




(a) The game (b) Nash equilibrium

Figure 3.1: A game.

$\mathrm{Pth} \to \mathbb{R}$ to denote the payoff function of Agent $\alpha_i$. That is, $\mathsf{payoff}_i$ assigns to each possible path $\pi$ a real number $\mathsf{payoff}_i(\pi)$ expressing the payoff of $\alpha_i$ on $\pi$. For a strategy profile $\mathsf{P}$, we use $\mathsf{payoff}_i(\mathsf{P})$ to abbreviate $\mathsf{payoff}_i(\mathsf{play}(\mathsf{P}, s_0))$. In the case of an LTL goal $\psi_i$, we define $\mathsf{payoff}_i(\pi) = 1$ if $\pi \models \psi_i$ and $\mathsf{payoff}_i(\pi) = 0$, otherwise.

The simplest and most appealing solution concept is dominant-strategies solution [OR94]. A *dominant strategy* is a strategy that an agent can never lose by adhering to, regardless of the strategies of the other agents. Therefore, if there is a profile of strategies $\mathsf{P} = \langle \mathsf{f}_0, \ldots, \mathsf{f}_k \rangle$ in which all strategies $\mathsf{f}_i$ are dominant, then no agent has an incentive to deviate from the strategy assigned to him in $\mathsf{P}$. Formally, $\mathsf{P}$ is a *dominant strategy profile* if for every $1 \leq i \leq k$ and for every (other) profile $\mathsf{P}'$, we have that $\mathsf{payoff}_i(\mathsf{P}') \leq \mathsf{payoff}_i(\mathsf{P}'[i \leftarrow \mathsf{f}_i])$.

As an example, consider the game in Figure 3.1, played by three agents, Alice, Bob, and Charlie, whose actions are $\{\mathtt{a}_1, \mathtt{a}_2\}$, $\{\mathtt{b}_1, \mathtt{b}_2\}$, and $\{\mathtt{c}_1, \mathtt{c}_2\}$, respectively. The arrows are labeled with the possible action of the agents. Each agent wants to visit a state marked with his initial letter, infinitely often. In this game, the strategy for Bob of always choosing $\mathtt{b}_2$ on his node 2 is dominant, while all the possible strategies for Charlie are dominant. On the other hand, Alice has no dominant strategies, since her goal essentially depends on the strategies adopted by the other agents. In several games, it can happen that agents have not any dominant strategy. For this reason, one would consider also other kind of solution concepts.

Another well known solution concept is Nash equilibrium [OR94]. A strategy profile is a *Nash equilibrium* if no agent has an incentive to deviate from his strategy in $\mathsf{P}$ provided that the other agents adhere to the strategies assigned to them in $\mathsf{P}$. Formally, $\mathsf{P}$ is a *Nash equilibrium profile* if for every $1 \leq i \leq k$ and for every (other) strategy $\mathsf{f}'_i$ for agent $\alpha_i$, we have that $\mathsf{payoff}_i(\mathsf{P}[i \leftarrow \mathsf{f}'_i]) \leq \mathsf{payoff}_i(\mathsf{P})$. An important advantage of Nash equilibrium is that it is more likely to exist than an equilibrium of dominant strategies [OR94] [1]. A weakness of Nash equilibrium is that it is not nearly as stable as a dominant-strategy solution: if one of the other agents deviates from his assigned strategy, nothing is guaranteed.

For the case of repeated-turn games like infinite games, a suitable refinement of Nash Equilibria is the *Subgame perfect Nash-equilibrium* [Sel75] (SPE, for short). A strategy profile $\mathsf{P} = \langle \mathsf{f}_0, \ldots, \mathsf{f}_k \rangle$ is an SPE if for every possible history of the game, no agent $\alpha_i$ has an incentive to deviate from her strategy $\mathsf{f}_i$, assuming that the other agents follow their strategies in $\mathsf{P}$. Intuitively, an SPE requires the existence of a Nash Equilibrium for each subgame starting from a randomly generated finite path of the original one. In [FKL10], the authors have studied cooperative rational synthesis also for the solution concept of SPE. To

[1] In particular, all $k$-agent turn-based games with $\omega$-regular objectives have Nash equilibrium [CMJ04].

do this, the synthesis algorithm in [FKL10] was extended to consider all possible histories of the game. In SL such a path property can be expressed combining strategy quantifiers with temporal operators. Indeed, the formula $\varphi = [\![\vec{x}]\!]\flat(\vec{x})\mathtt{G}\psi(\vec{y})$, with $\mathsf{free}(\varphi) = \vec{y}$, states that, for all possible profile strategies the agents can follow, the game always reaches a position in which the formula $\psi(\vec{y})$ holds. Thus, for all possible paths that can be generated by agents, the property holds. By replacing $\psi(\vec{y})$ with the above formula, we then obtain a formula that represents SPEs. Hence, the cooperative and non-cooperative synthesis problems can be asserted in SL also for SPE, and our results hold also for this solution concept.

In rational synthesis, we control the strategy of the system and assume that the agents that constitute the environment are rational. Consider a strategy profile $\mathsf{P} = \langle \mathsf{f}_0, \ldots, \mathsf{f}_k \rangle$ and a solution concept $\gamma$ (that is, $\gamma$ is either "dominant strategies" or "Nash equilibrium"). We say that $\mathsf{P}$ is *correct* if $\mathsf{play}(\mathsf{P})$ satisfies $\varphi_0$. We say that $\mathsf{P}$ is in a $\mathsf{f}_0$*-fixed* $\gamma$*-equilibrium* if the agents composing the environment have no incentive to deviate from their strategies according to the solution concept $\gamma$, assuming that the system continues to follow $\mathsf{f}_0$. Thus, $\mathsf{P}$ is a $\mathsf{f}_0$-fixed dominant-strategy equilibrium if for every $1 \leq i \leq k$ and for every (other) profile $\mathsf{P}'$ in which Agent 0 follows $\mathsf{f}_0$, we have that $\mathsf{payoff}_i(\mathsf{P}') \leq \mathsf{payoff}_i(\mathsf{P}'[i \leftarrow \mathsf{f}_i])$. Note that for the case of Nash equilibrium, adding the requirement that $\mathsf{P}$ is $\mathsf{f}_0$-fixed does not change the definition of an equilibrium.

In the context of objectives in LTL, we assume the following simple payoffs. If the objective $\varphi_i$ of Agent $\alpha_i$ holds, then his payoff is 1, and if $\varphi_i$ does not hold, then the payoff of Agent $i$ is 0. Accordingly, $\mathsf{P} = \langle \mathsf{f}_0, \ldots, \mathsf{f}_k \rangle$ is in a dominant-strategy equilibrium if for every $1 \leq i \leq k$ and profile $\mathsf{P}' = \langle \mathsf{f}_0', \ldots, \mathsf{f}_k' \rangle$ with $\mathsf{f}_0' = \mathsf{f}_0$, if $\mathsf{play}(\mathsf{P}') \models \varphi_i$, then $\mathsf{play}(\mathsf{P}'[i \leftarrow \mathsf{f}_i]) \models \varphi_i$. Also, $\mathsf{P}$ is in a Nash-equilibrium if for every $1 \leq i \leq k$ and strategy $\mathsf{f}_i'$, if $\mathsf{play}(\mathsf{P}[i \leftarrow \mathsf{f}_i']) \models \varphi_i$, then $\mathsf{play}(\mathsf{P}) \models \varphi_i$.

**Definition 3.1.1 (Rational synthesis)** *The input to the rational-strategy problem is a set* $\mathrm{X}$ *of atomic propositions, partitioned into* $\mathrm{X}_0, \ldots, \mathrm{X}_k$*,* LTL *formulas* $\varphi_0, \ldots, \varphi_k$*, describing the objectives of the system and the agents composing the environment, and a solution concept* $\gamma$*. We distinguish between two variants of the problem:*

1. *In* Cooperative rational synthesis [FKL10]*, the desired output is a strategy profile* $\mathsf{P}$ *such that* $\mathsf{play}(\mathsf{P})$ *satisfies* $\varphi_0$ *and* $\mathsf{P}$ *is a* $\mathsf{f}_0$*-fixed* $\gamma$*-equilibrium.*

2. *In* Non-cooperative rational synthesis*, the desired output is a strategy* $\mathsf{f}_0$ *for the system such that for every strategy profile* $\mathsf{P}$ *that includes* $\mathsf{f}_0$ *and is a* $\mathsf{f}_0$*-fixed* $\gamma$*-equilibrium, we have that* $\mathsf{play}(\mathsf{P})$ *satisfies* $\varphi_0$*.*

Thus, in the cooperative variant of [FKL10], we assume that once we suggest to the agents in the environment strategies that are in a $\gamma$-equilibrium, they will adhere to the suggested strategies. In the non-cooperative variant we introduce here, the agents may follow any strategy profile that is in a $\gamma$-equilibrium, and thus we require the outcome of all these profiles to satisfy $\varphi_0$. It is shown in [FKL10] that the cooperative rational synthesis problem is 2EXPTIME-COMPLETE.

Note that the input to the rational synthesis problem may not have a solution, so when we solve the rational-synthesis problem, we first solve the *rational-realizability* problem,

which asks if a solution exists. As with classical synthesis, the fact that SL model-checking algorithms can be easily modified to return a regular witness for the involved strategies in case an existentially quantified strategy exists, makes the realizability and synthesis problems strongly related.

**Example 3.1.1** *Consider a file-sharing network with the system and an environment consisting of two agents. The system controls the signal* $\mathtt{d}_1$ *and* $\mathtt{d}_2$ *(Agent* $\alpha_1$ *and* $\alpha_2$ *can download, respectively) and it makes sure that an agent can download only when the other agent uploads. The system's objective is that both agents will upload infinitely often. Agent* $\alpha_1$ *controls the signal* $\mathtt{u}_1$ *(Agent* $\alpha_1$ *uploads), and similarly for Agent* $\alpha_2$ *and* $\mathtt{u}_2$*. The goal of both agents is to download infinitely often.*

*Formally, the set of atomic propositions is* $\mathrm{X} = \{\mathtt{d}_1, \mathtt{d}_2, \mathtt{u}_1, \mathtt{u}_2\}$*, partitioned into* $\mathrm{X}_0 = \{\mathtt{d}_1, \mathtt{d}_2\}$, $\mathrm{X}_1 = \{\mathtt{u}_1\}$*, and* $\mathrm{X}_2 = \{\mathtt{u}_2\}$*. The objectives of the system and the environment are as follows.*

- $\varphi_0 = \mathtt{G}(\neg \mathtt{u}_1 \rightarrow \neg \mathtt{d}_2) \wedge \mathtt{G}(\neg \mathtt{u}_2 \rightarrow \neg \mathtt{d}_1) \wedge \mathtt{GF}\mathtt{u}_1 \wedge \mathtt{GF}\mathtt{u}_2$,
- $\varphi_1 = \mathtt{GF}\mathtt{d}_1$,
- $\varphi_2 = \mathtt{GF}\mathtt{d}_2$.

*First, note that in standard synthesis,* $\varphi_0$ *is not realizable, as a hostile environment needs not upload. In the cooperative setting, the system can suggest to both agents the following* TIT FOR TAT *strategy: upload at the first time step, and from that point onward upload iff the other agent uploads. The system itself follows a strategy* $\mathsf{f}_0$ *according to which it enables downloads whenever possible (that is,* $\mathtt{d}_2$ *is valid whenever Agent* $\alpha_1$ *uploads, and* $\mathtt{d}_1$ *is valid whenever Agent* $\alpha_2$ *uploads). It is not hard to see that the above three strategies are all dominant. Indeed, all the three objectives are satisfied. Thus, the triple of strategies is a solution for the cooperative setting, for both solution concepts.*

*What about the non-cooperative setting? Consider the above strategy* $\mathsf{f}_0$ *of the system, and consider strategies for the agents that never upload. The tuple of the three strategies is in a* $\mathsf{f}_0$*-fixed Nash equilibrium.*

*This ensures strategies for the environment to be dominant. Indeed, if Agent* $\alpha_2$ *changes her strategy,* $\varphi_1$ *is still satisfied and vice-versa. Indeed, as long as Agent* $\alpha_2$ *sticks to his strategy, Agent* $\alpha_1$ *has no incentive to change his strategy, and similarly for Agent* $\alpha_2$*. Thus,* $\mathsf{f}_0$ *is not a solution to the non-cooperative rational synthesis problem for the solution concept of Nash equilibrium. On the other hand, we claim that* $\mathsf{f}_0$ *is a solution to the non-cooperative rational synthesis problem for the solution concept of dominant strategies. For showing this, we argue that if* $\mathsf{f}_1$ *and* $\mathsf{f}_2$ *are dominant strategies for* $\alpha_1$ *and* $\alpha_2$*, then* $\varphi_0$ *is satisfied in the path that is the outcome of the profile* $\mathsf{P} = \langle \mathsf{f}_0, \mathsf{f}_1, \mathsf{f}_2 \rangle$*. To see this, consider such a path* $\pi = \mathsf{play}(\mathsf{f}_0, \mathsf{f}_1, \mathsf{f}_2)$*. We necessarily have that* $\pi \models \varphi_1 \wedge \varphi_2$*. Indeed, otherwise* $\mathsf{f}_1$ *and* $\mathsf{f}_2$ *would not be dominant, as we know that with the strategies described above,* $\alpha_1$ *and* $\alpha_2$ *can satisfy their objectives. Now, since* $\pi \models \varphi_1 \wedge \varphi_2$*, we know that* $\mathtt{u}_2$ *and* $\mathtt{u}_1$ *hold infinitely often in* $\pi$*. Also, it is not hard to see that the formulas* $\mathtt{G}(\neg \mathtt{u}_1 \rightarrow \neg \mathtt{d}_2)$ *and* $\mathtt{G}(\neg \mathtt{u}_2 \rightarrow \neg \mathtt{d}_1)$ *are always satisfied in the context of* $\mathsf{f}_0$*, no matter how the other agents behave. It follows that* $\pi \models \varphi_0$*, thus* $\mathsf{f}_0$ *is a solution of the non-cooperative rational synthesis problem for dominant strategies.*

## 3.2 Qualitative Rational Synthesis

In this section we study cooperative and non-cooperative rational synthesis and show that they can be reduced to the model-checking problem for SL[NG]. The cooperative and non-cooperative rational synthesis problems for several solution concepts can be stated in SL[NG].

We first show how to state that a given strategy profile $\vec{y} = (y_0, \ldots, y_k)$ is in a $y_o$-fixed $\gamma$-equilibrium. For $\alpha_i \in \mathrm{Ag}$, let $\varphi_i$ be the objective of Agent $\alpha_i$. Moreover, for a given strategy profile $\vec{y}$, by $\flat(\vec{y}) = (\alpha, y_o) \ldots (\alpha, y_n)$ we denote the binding prefix assigning each strategy to the corresponding agent, and by $\flat(\vec{y}_{-i}, z_i)]) = (\alpha, y_o) \ldots (\alpha, y_{i-1})(\alpha, z_i)(\alpha, y_{i+1}) \ldots (\alpha, y_n)$ we denote the binding prefix assigning the each strategy to the corresponding agent but $\alpha$, to which it assigns the strategy $z_i$. For a solution concept $\gamma$ and a strategy profile $\vec{y} = (y_0, \ldots, y_k)$, the formula $\varphi^\gamma(\vec{y})$, expressing that the profile $\vec{y}$ is a $y_0$-fixed $\gamma$-equilibrium, is defined as follows.

- For the solution concept of dominant strategies, we define:
  $\varphi^\gamma(\vec{y}) := [\![\vec{z}]\!] \bigwedge_{i=1}^{k} (\flat(\vec{z})\varphi_i \rightarrow \flat(\vec{z}_{-i}, y_i)])\varphi_i)$.
- For the solution concept of Nash equilibrium, we define:
  $\varphi^\gamma(\vec{y}) := [\![\vec{z}]\!] \bigwedge_{i=1}^{k} (\flat(\vec{y}_{-i}, z_i)\varphi_i \rightarrow \flat(\vec{y})\varphi_i)$.
- For the solution concept of Subgame Perfect Equilibrium, we define:
  $\varphi^\gamma(\vec{y}) := [\![\vec{x}]\!]\flat(\vec{x}_{-0}, y_o)\mathsf{F} \bigwedge_{i=1}^{k} [\![z_i]\!]\flat(\vec{y}_{-i}, z_i)\varphi_i \rightarrow \flat(\vec{y})\varphi_i$.

We can now state the existence of a solution to the cooperative and non-cooperative rational-synthesis problem, respectively, with input $\varphi_0, \ldots, \varphi_k$ by the closed formulas:

1. $\varphi^\gamma_{cRS} := \langle\!\langle y_o \rangle\!\rangle \langle\!\langle y_1 \rangle\!\rangle \ldots \langle\!\langle y_k \rangle\!\rangle (\varphi^\gamma(\vec{y}) \wedge \varphi_0)$;
2. $\varphi^\gamma_{noncRS} := \langle\!\langle y_o \rangle\!\rangle [\![y_1]\!] \ldots [\![y_k]\!] (\varphi^\gamma(\vec{y}) \rightarrow \varphi_0)$.

Indeed, the formula 1 specifies the existence of a strategy profile $\mathsf{P} = \langle \mathsf{f}_o, \ldots, \mathsf{f}_k \rangle$ that is $\mathsf{f}_o$-fixed $\gamma$-equilibrium and such that the outcome satisfies $\varphi_0$. On the other hand, the formula 2 specifies the existence of a strategy $\mathsf{f}_o$ for the system such that the outcome of all profiles that are in a $\mathsf{f}_o$-fixed $\gamma$-equilibrium satisfy $\varphi_0$.

As shown above, all the solution concepts we are taking into account can be specified in SL[NG] with formulas whose length is polynomial in the number of the agents and in which the alternation depth of the quantification is 1. Hence we can apply the known complexity results for SL[NG]:

**Theorem 3.2.1 (Cooperative and non-cooperative rational-synthesis complexity)** *The cooperative and non-cooperative rational-synthesis problems in the qualitative setting are* 2EXPTIME*-complete.*

*Proof.* Consider an input $\varphi_0, \ldots, \varphi_k$, X, and $\gamma$ to the cooperative or non-cooperative rational-synthesis problem. As explained in Section 3.1, the input induces a game $\mathcal{G}$ with nodes in $2^X$. As detailed above, there is a solution to the cooperative (resp., non-cooperative)

problem iff the SL[NG] formula $\varphi^{\gamma}_{cRS}$ (resp., $\varphi^{\gamma}_{noncRS}$) is satisfied in $\mathcal{G}$. The upper bound then follows from the fact that the model checking problem for SL[NG] formulas of alternation depth 1 is in 2EXPTIME in the size of the formula (Cf., Theorem 2.3.4 on page 47). Moreover, the model-checking algorithm can return finite-state transducers that model strategies that are existentially quantified.

For the lower bound, it is easy to see that the classical LTL synthesis problem is a special case of the cooperative and non-cooperative rational synthesis problem. Indeed, $\varphi(I, O)$ is realizable against a hostile environment iff the solution to the non-cooperative rational synthesis problem for a system that has an objective $\varphi$ and controls $I$ and an environment that consists of a single agent that controls $O$ and has an objective *True*, is positive. □

## 3.3 Quantitative Rational Synthesis

As discussed in this chapter introduction, a weakness of classical synthesis algorithms is the fact that specifications are Boolean and miss a reference to the quality of the satisfaction. Applications of game theory consider games with quantitative payoffs. Thus, even more than the classical setting, the rational setting calls for an extension of the synthesis problem to a quantitative setting. In this section we introduce *Objective* LTL, a quantitative extension of LTL, and study an extension of the rational-synthesis problem for specifications in Objective LTL. As opposed to other multi-valued specification formalisms used in the context of synthesis of high-quality systems [BCHJ09, ABK13], Objective LTL uses the syntax and semantics of LTL and only augments the specification with a reward function that enables a succinct and convenient prioritization of sub-specifications.

*Objective* LTL (OLTL, for short) is an extension of LTL in which specifications consist of sets of LTL formulas weighted by functions. Formally, an OLTL *specification* over a set X of atomic propositions is a pair $\theta = \langle \Psi, \mathsf{f} \rangle$, where $\Psi = \langle \psi_1, \psi_2, \ldots, \psi_m \rangle$ is a tuple of LTL formulas over X and $\mathsf{f} : \{0,1\}^m \to \mathbb{Z}$ is a *reward function*, mapping Boolean vectors of length $m$ to integers. We assume that $\mathsf{f}$ is given by a polynomial. We use $|\Psi|$ to denote $\sum_{i=1}^{m} |\psi_i|$. For a computation $\pi \in (2^{\mathrm{X}})^{\omega}$, the *signature of* $\Psi$ *in* $\pi$, denoted $\mathsf{sig}(\Psi, \pi)$, is a vector in $\{0,1\}^m$ indicating which formulas in $\Psi$ are satisfied in $\pi$. Thus, $\mathsf{sig}(\Psi, \pi) = \langle v_1, v_2, \ldots, v_m \rangle$ is such that for all $1 \leq i \leq m$, we have that $v_i = 1$ if $\pi \models \psi_i$ and $v_i = 0$ if $\pi \not\models \psi_i$. The *value of* $\theta$ *in* $\pi$, denoted $\mathsf{val}(\theta, \pi)$ is then $\mathsf{f}(\mathsf{sig}(\Psi, \pi))$. Thus, the interpretation of an OLTL specification is quantitative. Intuitively, $\mathsf{val}(\theta, \pi)$ indicates the level of satisfaction of the LTL formulas in $\Psi$ in $\pi$, as determined by the priorities induced by $\mathsf{f}$. We note that the input of weighted LTL formulas studied in [CY98] is a special case of Objective LTL.

**Example 3.3.1** *Consider a system with $m$ buffers, of capacities $\mathtt{c}_1, \ldots, \mathtt{c}_m$. Let $\mathtt{full}_i$, for $1 \leq i \leq m$, indicate that buffer $i$ is full. The* OLTL *specification $\theta = \langle \Psi, \mathsf{f} \rangle$, with $\Psi = \langle \mathtt{Ffull}_1, \mathtt{Ffull}_2, \ldots, \mathtt{Ffull}_m \rangle$ and $\mathsf{f}(\mathtt{v}) = \mathtt{c}_1 \cdot \mathtt{v}_1 + \cdots + \mathtt{c}_m \cdot \mathtt{v}_m$ enables us to give different satisfaction values to the objective of filling a buffer. Note that $\mathsf{val}(\langle \Psi, \mathsf{f} \rangle, \pi)$ in a computation $\pi$ is the sum of capacities of all filled buffers.*

In the quantitative setting, the objective of Agent $\alpha$ is given by means of an OLTL specification $\theta_i = \langle \Psi_i, \mathsf{f}_i \rangle$, specifications describe the payoffs to the agents, and the objective

of each agent (including the system) is to maximize his payoff. For a strategy profile $\mathsf{P}$, the payoff for Agent $\alpha$ in $\mathsf{P}$ is simply $\mathsf{val}(\theta_i, \mathsf{play}(\mathsf{P}))$.

In the quantitative setting, rational synthesis is an optimization problem. Here, in order to easily solve it, we provide a decision version by making use of a threshold. It is clear that the optimization version can be solved by searching for the best threshold in the decision one.

**Definition 3.3.1 (Quantitative rational synthesis)** *The input to the quantitative rational-strategy problem is a set* $\mathrm{X}$ *of atomic propositions, partitioned into* $\mathrm{X}_0, \ldots, \mathrm{X}_k$*,* OLTL *specifications* $\theta_0, \theta_1, \ldots, \theta_k$*, with* $\theta_i = \langle \Psi_i, \mathsf{f}_i \rangle$*, and a solution concept* $\gamma$*. We distinguish between two variants of the problem:*

1. *In* cooperative quantitative rational synthesis*, the desired output for a given threshold* $t \in \mathbb{N}$ *is a strategy profile* $\mathsf{P}$ *such that* $\mathsf{payoff}_0(\mathsf{P}) \geq t$ *and* $\mathsf{P}$ *is in a* $\mathsf{f}_\mathrm{o}$*-fixed* $\gamma$*-equilibrium.*

2. *In* non-cooperative quantitative rational synthesis*, the desired output for a given threshold* $t \in \mathbb{N}$ *is a strategy* $\mathsf{f}_\mathrm{o}$ *for the system such that, for each strategy profile* $\mathsf{P}$ *including* $\mathsf{f}_\mathrm{o}$ *and being in a* $\mathsf{f}_\mathrm{o}$*-fixed* $\gamma$*-equilibrium, we have that* $\mathsf{payoff}_0(\mathsf{P}) \geq t$*.*

Now, we introduce some auxiliary formula that helps us to formulate also the quantitative rational-synthesis problem in SL[NG].

For a tuple $\Psi = \langle \psi_1, \ldots, \psi_m \rangle$ of LTL formulas and a signature $v = \{v_1, \ldots, v_m\} \in \{0,1\}^m$, let $\mathsf{mask}(\Psi, v)$ be an LTL formula that characterizes computations $\pi$ for which $\mathsf{sig}(\Psi, \pi) = v$. Thus, $\mathsf{mask}(\Psi, v) = (\bigwedge_{i:v_i=0} \neg\psi_i) \wedge (\bigwedge_{i:v_i=1} \psi_i)$.

We adjust the SL formulas $\Phi^\gamma(\vec{y})$ described in Section 3.2 on page 65 to the quantitative setting. Recall that $\Phi^\gamma(\vec{y})$ holds iff the strategy profile assigned to $\vec{y}$ is in a $\mathsf{f}_\mathrm{o}$-fixed $\gamma$-equilibrium. There, the formula is a conjunction over all agents in $\{\alpha, \ldots, \alpha\}$, stating that Agent $\alpha$ does not have an incentive to change his strategy. In our quantitative setting, this means that the payoff of Agent $\alpha$ in an alternative profile is not bigger than his payoff in $\vec{y}$. For two strategy profiles, assigned to $\vec{y}$ and $\vec{y}'$, an SL formula that states that Agent $\alpha$ has no incentive that the profile would change from $\vec{y}$ to $\vec{y}'$ can state that the signature of $\Psi$ in $\mathsf{play}(\vec{y}')$ results in a payoff to Agent $\alpha$ that is smaller than his current payoff. Formally, we have that:

$$\Phi_i^{eq}(\vec{y}, \vec{y}') = \bigvee_{v \in \{1,\ldots,m_i\} : \mathsf{f}_i(v) \leq \mathsf{payoff}_i(\vec{y})} \flat(\vec{y}')\mathsf{mask}(\Psi_i, v).$$

We can now adjust $\Phi^\gamma(\vec{y})$ for all the cases of solution concepts we are taking into account.

- For the solution concept of dominant strategies, we define:
  $\Phi^\gamma(\vec{y}) := \bigwedge_{i \in \{1,\ldots,n\}} [\![\vec{z}]\!] \Phi_i^{eq}(\vec{y}, (\vec{z}[\alpha \leftarrow y_\mathrm{o}])$;
- For the solution concept of Nash equilibrium, we define:
  $\Phi^\gamma(\vec{y}) := \bigwedge_{i \in \{1,\ldots,n\}} [\![\vec{z}]\!] \Phi_i^{eq}(\vec{y}, (\vec{y}[\alpha \leftarrow z_i])$;
- For the solution concept of Subgame Perfect Equilibrium, we define:
  $\varphi^\gamma(\vec{y}) := [\![\vec{x}]\!] \flat(\vec{x}[\alpha \leftarrow y_\mathrm{o}]) \mathtt{F} \bigwedge_{i \in \{1,\ldots,n\}} [\![\vec{z}]\!] \Phi_i^{eq}(\vec{y}, (\vec{z}[\alpha \leftarrow y_0]))$.

Once we adjust $\Phi^{\gamma}(\vec{y})$ to the quantitative setting, we can use the same SL formula used in the non-quantitative setting to state the existence of a solution to the rational synthesis problem. We have the following:

- $\Phi^{\gamma}_{RS} := \langle\langle y_0 \rangle\rangle \langle\langle y_1 \rangle\rangle \ldots \langle\langle y_k \rangle\rangle (\Phi^{\gamma}(\vec{y}) \wedge \varphi_0)$;
- $\Phi^{\gamma}_{nonRS} := \langle\langle y_0 \rangle\rangle [\![ y_1 ]\!] \ldots [\![ y_k ]\!] (\Phi^{\gamma}(\vec{y}) \rightarrow \varphi_0)$.

**Theorem 3.3.1** *The cooperative and non-cooperative quantitative rational-synthesis problems are 2*EXPTIME-COMPLETE.

*Proof.* We can reduce the problems to the model-checking problem of the SL formulas $\Phi^{\gamma}_{RS}$ and $\Phi^{\gamma}_{nonRS}$, respectively. We should, however, take care when analyzing the complexity of the procedure, as the formulas $\Phi^{eq}_{i}(\vec{y}, \vec{y}')$, which participate in $\Phi^{\gamma}_{RS}$ and $\Phi^{\gamma}_{nonRS}$ involve a disjunction over vectors in $\{0,1\}^{m_i}$, resulting in $\Phi^{\gamma}_{nonRS}$ of an exponential length.

While the above prevents us from using the doubly exponential known bound on SL model checking for formulas of alternation depth 1 as is, it is not difficult to observe that the run time of the model-checking algorithm in [MMPV14], when applied to $\Phi^{\gamma}_{nonRS}$, is only doubly exponential. The reason is the fact that the inner exponent paid in the algorithm for SL model checking is due to the blow-up in the translation of the formula to a nondeterministic Büchi automaton over words (NBW, for short). In this translation, the exponentially many disjuncts are dominated by the exponential translation of the innermost LTL to NBW. Thus, the running time of the algorithm is doubly exponential, and it can return the witnessing strategies.

Hardness in 2EXPTIME follows easily from hardness in the non-quantitative setting.

□

CHAPTER 4

# Pushdown ATL$^\star$

The chapter is organized as follows. In Section 4.1 we introduce the definition of PGS, where we also provide an example that helps clarifying the setting. There, we also show how a PGS can be embedded into a (infinite-state) CGS. In Section 4.2 we recall the syntax and the semantics of ATL$^\star$ over CGSs and then, exploiting the results in the previous section, we define the model-checking problem of ATL$^\star$ over PGSs. Finally, in Section 4.3 we show that this problem can be solved in 3EXPTIME by means of an automata-theoretic approach. There, we also show a 2EXPSPACE-HARD lower-bound.

## 4.1 Pushdown Games

Classically, ATL$^\star$ formulas are interpreted over *Concurrent Game Structures* [AHK02]. Here, we instead interpret ATL$^\star$ formulas over a new semantic framework, which we call *Pushdown Game Structure*. Intuitively, this new formalism provides a concurrent game structure in which a stack is added and the labeling and transition functions depend on its content. In this section, we also show that every pushdown game structure can be transformed into a suitable concurrent game structure, so providing the required interpretation of ATL$^\star$ formulas over the former. However, note that the latter requires a infinite number of states, used to represent all the possible configurations the pushdown system can enter. We start with the definition of pushdown game structures.

**Definition 4.1.1 (Pushdown Game Structure)** *A* Pushdown Game Structure (PGS, for short) *is a tuple* $\mathcal{P} = \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}, \mathrm{Loc}, \Gamma, \mathsf{tr}, \mathsf{ap}, l_{\mathrm{o}} \rangle$*, where* AP*,* Ag*,* Ac*,* Loc *and* $\Gamma$ *are finite sets of atomic propositions, agents, actions, locations, and stack alphabet, respectively,* $l_{\mathrm{o}} \in \mathrm{Loc}$ *is an initial location, and* $\mathsf{ap} : \mathrm{Loc} \times \Gamma_\bot \to 2^{\mathrm{AP}}$ *is a labeling function, where* $\Gamma_\bot = \Gamma \cup \{\bot\}$ *and* $\bot$ *is the special* bottom *stack symbol not contained in* $\Gamma$*. Let* $\mathrm{Dc} \triangleq \mathrm{Ac}^{\mathrm{Ag}}$ *be the set of* decisions*,* i.e.*, functions from* Ag *to* Ac *representing the action choices for each agent. Then,* $\mathsf{tr} : \mathrm{Loc} \times \Gamma_\bot \times \mathrm{Dc} \to \mathrm{Loc} \times \Gamma_\bot^*$ *is a* transition function *mapping a location, a stack symbol, and a decision to a location and a word in the stack alphabet.*

A pair $s = (l, \alpha) \in \mathrm{St} \triangleq \mathrm{Loc} \times (\Gamma^* \cdot \{\bot\})$ is called *state* or *configuration*. We write $\mathsf{top}(\alpha)$ for the left most symbol of $\alpha$ and call it the *top* of the stack content $\alpha$. The PGS moves according to the transition function. This means that, if it is in the location $l$, the top of the stack content is $\gamma$, and the agents make a decision $\delta$, then $\mathsf{tr}(l, \gamma, \delta) = (l', \alpha)$ means that the execution moves to the location $l'$ and the symbol $\gamma$ is replaced with $\alpha$ on the top of the stack content. We assume that, if $\bot$ is popped, then it is pushed right back, and that is the only case in which it is pushed. This means that $\bot$ is always on the bottom of the stack and nowhere else. The stack containing only the symbol $\bot$ is said to be *empty*.

As the stack has no a priori bound on its size, the set $\mathrm{St}$ is assumed to be possibly infinite. Saying this, it turns out that PGSs are *infinite-state multi-agent systems*.

The notion of labeling and transition can be lifted to states, as follows. For a state $s = (l, \alpha)$, we define $\mathsf{ap}(s) = \mathsf{ap}(l, \mathsf{top}(\alpha))$. Moreover, for a decision $\delta \in \mathrm{Dc}$, we define $\mathsf{tr}((l, \gamma \cdot \alpha), \delta) = (l', \beta \cdot \alpha)$, with $(l', \beta) = \mathsf{tr}(l, \gamma, \delta)$.

Note that for a classical *pop* we write the empty word $\varepsilon$ on the stack. To make a classical *push* one has to first put back the read top symbol and then push the required word. The transition function also allows to perform in one step a *pop-push* operation that replaces the top stack symbol with the required word. To get familiar with PGSs, we give an example.

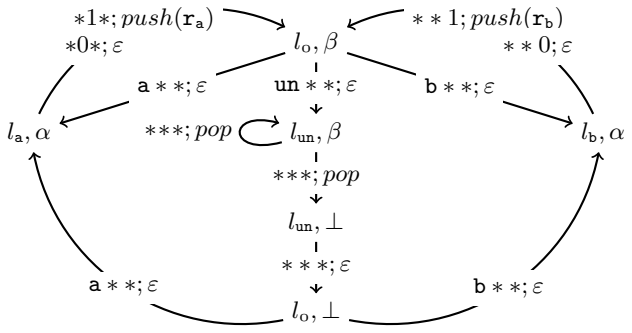



Figure 4.1: A Pushdown system scheduler.

**Example 4.1.1 (Pushdown scheduler)** *Take a system consisting of two processes* $\mathtt{a}$ *and* $\mathtt{b}$ *that may access to a common resource via the respective requests* $\mathtt{r_a}$ *and* $\mathtt{r_b}$ *and a scheduler* $\mathtt{s}$ *that can grant in a* LIFO *order the processes requests, all memorized into a stack. As model we use a* PGS $\mathcal{P} = \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}, \mathrm{Loc}, \Gamma, \mathsf{tr}, \mathsf{ap}, l_o \rangle$*, with* $\mathrm{AP} = \{\mathtt{r_a}, \mathtt{r_b}, \mathtt{g_a}, \mathtt{g_b}, \mathtt{a}, \mathtt{b}, \mathtt{e}\}$, $\mathrm{Ag} = \{\mathtt{s}, \mathtt{a}, \mathtt{b}\}$, $\mathrm{Ac} = \{\mathtt{a}, \mathtt{b}, \mathtt{un}, 0, 1\}$, $\mathrm{Loc} = \{l_o, l_\mathtt{a}, l_\mathtt{b}, l_\mathtt{un}\}$*, and* $\Gamma = \{\mathtt{r_a}, \mathtt{r_b}\}$*. The scheduler* $\mathtt{s}$ *controls the location* $l_o$ *by means of the actions* $\mathtt{a}$*,* $\mathtt{b}$*, and* $\mathtt{un}$*, standing for "*$\mathtt{a}$ *can make a request", "*$\mathtt{b}$ *can make a request", and "the system can unload the stack requests", respectively. Accordingly, they lead to the locations* $l_\mathtt{a}$*,* $l_\mathtt{b}$*, and* $l_\mathtt{un}$*. On* $l_\mathtt{a}$ *and* $l_\mathtt{b}$*, the agents* $\mathtt{a}$ *and* $\mathtt{b}$*, respectively, can either make a request via action* $1$ *or skip it with action* $0$*. In the former case, the request is recorded into the stack by writing the symbol* $\mathtt{r_x}$*, for* $\mathtt{x} \in \{\mathtt{a}, \mathtt{b}\}$*; otherwise, in the latter case there is no operation over the stack. Finally, the location* $l_\mathtt{un}$ *triggers the granting phase by emptying the stack. During this phase, neither* $\mathtt{a}$ *nor* $\mathtt{b}$ *can make any further request. This can be seen as a legitimate constraint by thinking how classical synchronizing and backup systems are designed.*

*The labeling function, for all* $\gamma \in \Gamma_\bot$ *and* $\mathtt{x} \in \{\mathtt{a}, \mathtt{b}\}$*, is defined as follows:* $\mathsf{ap}(l_o, \bot) = \emptyset$*,* $\mathsf{ap}(l_o, \mathtt{r_x}) = \{\mathtt{x}\}$*,* $\mathsf{ap}(l_\mathtt{x}, \gamma) = \{\mathtt{x}\}$*,* $\mathsf{ap}(l_\mathtt{un}, \mathtt{r_x}) = \{\mathtt{g_x}\}$*, and* $\mathsf{ap}(l_\mathtt{un}, \bot) = \{\mathtt{e}\}$*. Intuitively, propositions* $\mathtt{a}$ *and* $\mathtt{b}$ *means that agents* $\mathtt{a}$ *and* $\mathtt{b}$ *are authorized to make a request, respectively. The proposition* $\mathtt{r_x}$*, instead, occurs when the corresponding request has been just made by agent* $\mathtt{x}$*. On the other hand, the proposition* $\mathtt{g_x}$ *occurs when the request* $\mathtt{r_x}$ *has been just granted. Finally,* $\mathtt{e}$ *indicates that the unloading phase is terminated and so the stack is empty.*

*The transition function* $\mathtt{tr}$ *is described directly in Figure 4.1. The labeling of the edges have the following meaning. First, note that it is composed of two parts separated by a semi-column. The left part represents the decision of the agents, given in the order* $\mathtt{s} < \mathtt{a} < \mathtt{b}$*. The right part represents the stack operation. As an example, the label* $*1*; push(\mathtt{r_a})$ *says that agent* $\mathtt{a}$ *is making a request* $\mathtt{r_a}$ *and the symbol* $\mathtt{r_a}$ *is pushed on the stack, where the symbol* $*$ *denotes any possible action for the other agents. The nodes represent all possible states. Note that, for the locations* $l_\mathtt{a}$ *and* $l_\mathtt{b}$ *we have collapsed the two possible configurations*

*with $\beta = \bot$ and $\beta \neq \bot$ since the transition over them does not depend on the stack content. Finally, observe that the stack is unbounded and so an execution might generate an infinite number of distinguished states. Also, observe that the stack is fundamental to keep track of the order in which the requests appear.*

Clearly, a PGS $\mathcal{P} = \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}, \mathrm{Loc}, \Gamma, \mathtt{tr}, \mathtt{ap}, l_{\mathrm{o}} \rangle$ can be suitably turned into a CGS $\mathcal{G}_{\mathcal{P}} = \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}, \mathrm{St}, \mathtt{tr}, \mathtt{ap}, s_{\mathrm{o}} \rangle$, where $\mathrm{St} = \mathrm{Loc} \times \Gamma_{\bot}^{*}$, $s_{\mathrm{o}} = (l_{\mathrm{o}}, \bot)$, and the functions $\mathtt{ap}$ and $\mathtt{tr}$ are the lifting on states of the corresponding functions in $\mathcal{P}$. Intuitively, the states of $\mathcal{G}$ are used to implicitly represent both the current location and store the stack content. Despite this, it is important to observe that, while a PGS has a finite number of control locations, the corresponding CGS necessarily has an infinite number of control states, as the number of different stack contents is unbounded.

## 4.2 ATL$^\star$ over pushdown systems

In this section, we recall the syntax of ATL$^\star$ and introduce its semantics over PGS via its representation in terms of CGS (with infinite states). We start with the definition of ATL$^\star$ syntax.

**Definition 4.2.1 (ATL$^\star$ Syntax)** ATL$^\star$ *formulas are built inductively from the set of atomic propositions* AP *and agents* Ag*, by using the following grammar, where $p \in \mathrm{AP}$ and $\mathrm{A} \subseteq \mathrm{Ag}$:*

$$\varphi := p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \mathtt{X}\varphi \mid \varphi \mathtt{U} \varphi \mid \langle\!\langle \mathrm{A} \rangle\!\rangle \varphi.$$

A *sentence* is a Boolean combination of ATL$^\star$ formulas of the form $\langle\!\langle \mathrm{A} \rangle\!\rangle \psi$. Intuitively, $\langle\!\langle \mathrm{A} \rangle\!\rangle \psi$ means that each agent in A has a strategy such that, whatever the other agents do, the resulting play satisfies $\psi$.

We now provide some examples of ATL$^\star$ formulas that will be useful in the sequel. Precisely, we consider $\varphi_1 = \langle\!\langle \{\mathtt{s}\} \rangle\!\rangle (\mathtt{GFa} \wedge \mathtt{GFb} \wedge \mathtt{GFe})$, $\varphi_2 = \langle\!\langle \emptyset \rangle\!\rangle (\mathtt{GFa} \wedge \mathtt{GFb} \wedge \mathtt{GFe})$ and $\varphi_3 = \langle\!\langle \emptyset \rangle\!\rangle ((\mathtt{r_a} \wedge \mathtt{X}(\neg \mathtt{eU r_b})) \rightarrow (\mathtt{Fg_b} \rightarrow \mathtt{XFg_a}))$ over the sets AP and Ag given in Example 4.1.1 on the preceding page. The formula $\varphi_1$ states that agent $\mathtt{s}$ has a way to let propositions $\mathtt{a}$, $\mathtt{b}$, and $\mathtt{e}$ to occur infinitely often. The formula $\varphi_2$ states that, no matters how the agents behave, propositions $\mathtt{a}$, $\mathtt{b}$, and $\mathtt{e}$ occur infinitely often. Finally, the formula $\varphi_3$ states that, whenever a request $\mathtt{r_b}$ occurs after an $\mathtt{r_a}$ one in the same loading phase, then, if $\mathtt{r_b}$ is eventually granted, then $\mathtt{r_a}$ is granted later on as well.

We now provide the semantics of ATL$^\star$.

**Definition 4.2.2 (ATL$^\star$ Semantics)** *For a* CGS $\mathcal{G} = \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Ac}, \mathrm{St}, \mathtt{tr}, \mathtt{ap}, s_{\mathrm{o}} \rangle$ *and a path* $\pi \in \mathrm{Pth}$*, the modeling relation $\mathcal{G}, \pi \models \varphi$ is inductively defined as follows:*

- *The atomic and boolean cases are defined as usual;*
- $\mathcal{G}, \pi \models \langle\!\langle \mathrm{A} \rangle\!\rangle \varphi$ *if there is an assignment* $\chi_{\mathrm{A}}$ *such that* $\mathcal{G}, \pi' \models \varphi$*, for all* $\pi' \in \mathsf{play}(\chi_{\mathrm{A}}, (\pi)_0)$*;*
- $\mathcal{G}, \pi \models \mathtt{X}\varphi$ *if* $\mathcal{G}, (\pi)_{\geq 1} \models \varphi$*;*

- $\mathcal{G}, \pi \models \varphi_1 \mathtt{U} \varphi_2$ *if there exists* $j \in \mathbb{N}$ *such that* $\mathcal{G}, (\pi)_{\geq i} \models \varphi_1$*, for all* $i < j$*, and* $\mathcal{G}, (\pi)_{\geq j} \models \varphi_2$.

For a sentence $\varphi$ and two paths $\pi_1, \pi_2$ with $(\pi_1)_0 = (\pi_2)_0$, it holds that $\mathcal{G}, \pi_1 \models \varphi$ iff $\mathcal{G}, \pi_2 \models \varphi$. Indeed, according to the semantics of the existential quantification $\langle\langle \mathrm{A} \rangle\rangle \psi$, the only element of the path to take into account is the first one. For this reason, for a sentence $\varphi$ we write $\mathcal{G}, s \models \varphi$ if $\mathcal{G}, \pi \models \varphi$ for some $\pi \in \mathrm{Pth}(s)$. Finally, we say that $\mathcal{G}$ satisfies $\varphi$, and write $\mathcal{G} \models \varphi$, if $\mathcal{G}, s_0 \models \varphi$.

To get familiar with the semantics, consider the PGS $\mathcal{P}$ given in Example 4.1.1 on page 70 and the formulas $\varphi_1$, $\varphi_2$, and $\varphi_3$ given above. It is easy to see that $\mathcal{G}_\mathcal{P} \models \varphi_1$. Indeed, the strategy $\mathsf{f}_1$ that allows the scheduler to pick infinitely often the actions `a`, `b`, and `un`, makes all the generated paths to satisfy $\mathtt{GFa} \wedge \mathtt{GFb} \wedge \mathtt{GFe}$. On the other hand, it is easy to see that $\mathcal{G}_\mathcal{P} \not\models \varphi_2$. Indeed, the strategy $\mathsf{f}_2$ for `s` such that $\mathsf{f}_2(\rho) = \mathtt{a}$, for all $\rho \in \mathrm{Trk}$, never makes `b` and `e` to occur in the generated paths. Finally, the formula $\varphi_3$ simply states that, whatever the three agents in the system do, the unloading phases always grant the requests in the reverse order they occur.

**Definition 4.2.3 (Model-checking)** *For a given* PGS $\mathcal{P}$ *and an* ATL$^\star$ *formula* $\varphi$*, the* model-checking problem *is to decide whether* $\mathcal{G}_\mathcal{P} \models \varphi$.

## 4.3 Model Checking

In this section, we provide a 3ExpTime upper-bound and a 2ExpSpace lower-bound for the model-checking problem of ATL$^\star$ over PGS.

### 4.3.1 Upper-bound Complexity

For the upper bound we use an automata-theoretic approach. We start with some notation and the definition of nondeterministic pushdown automata. See [KPV02, KVW00] for more.

For a given set $\mathrm{D} \subseteq \mathbb{N}$, a D-tree T is a prefix closed subset of $\mathrm{D}^*$, *i.e.*, a set in which, if $\mathtt{x} \cdot \mathtt{d} \in \mathrm{T}$ then $\mathtt{x} \in \mathrm{T}$. The elements of T are called *nodes* and the empty word $\varepsilon$ is called the *root* of T. For $x \in \mathrm{T}$, the set of *children* of $x$ is $children(\mathrm{T}, x) \triangleq \{x \cdot i \in \mathrm{T} : i \in \mathrm{D}\}$. For $x, y \in \mathrm{T}$, we write $x \prec y$ to mean that $x$ is a proper prefix of $y$, *i.e.*, there exists $z \in \mathrm{D}^*$ such that $x \cdot z = y$. For $x \in \mathrm{T}$, a path in T from $x$ is a minimal set $\pi \subseteq \mathrm{T}$ such that $x \in \pi$ and for each $y \in \pi$ such that $children(\mathrm{T}, y) \neq \emptyset$, there exist exactly one node in $children(\mathrm{T}, y)$ belonging to $\pi$. For an alphabet $\Sigma$, a $\Sigma$-labeled tree is a pair $\mathcal{T} = \langle \mathrm{T}, \mathsf{V} \rangle$ where T is a tree and $\mathsf{V} : \mathrm{T} \to \Sigma$ maps each node in T to an element in $\Sigma$. In the following, we mainly consider $\Sigma$ to be the set power set $2^{\mathrm{AP}}$ of atomic propositions AP.

A *Nondeteriministic Pushdown Tree Automata* (PD-NTA, for short), over $\Sigma$-labeled trees, is a tuple $\mathcal{A} = \langle \Sigma, \Gamma, \mathrm{Q}, q_0, \alpha_0, \delta, \mathrm{F} \rangle$, where $\Sigma$ and $\Gamma$ are *finite* input and stack alphabet sets, Q is a finite set of states, $q_0$ is an initial state, $\alpha_0 \in \Gamma^* \cdot \bot$ is an initial stack content, $\delta : \mathrm{Q} \times \Sigma \times \Gamma_\bot \to 2^{2^{(\mathrm{Q} \times \Gamma^*)}}$ is a transition function such that, for all $(q, \sigma, \gamma) \in \mathrm{Q} \times \Sigma \times \Gamma_\bot$, $\delta(q, \sigma, \gamma)$ is a *finite* set, and F is an acceptance condition over Q.

When the automaton is in a state $q$, reading an input node $x$ labeled with $\sigma \in \Sigma$, and the stack contains a word $\gamma \cdot \alpha$ in $\Gamma_{\perp}^{*}$, it chooses, for some $k \in \mathbb{N}$, a tuple $\langle (q_1, \beta_1), \ldots, (q_k, \beta_k) \rangle \in \delta(q, \sigma, \gamma)$ and splits in $k$ copies such that, for each $1 \leq i \leq k$, the $i$-th copy is in the state $q_i$ and the stack content is updated by removing $\gamma$ and pushing $\beta_i$. Then, it reads the node $x \cdot i$.

A run of a PD-NTA on a $\Sigma$-labeled tree $\mathcal{T} = \langle \mathrm{T}, \mathsf{V} \rangle$ is a $(\mathrm{Q} \times \Gamma_{\perp}^{*})$-labeled tree $\langle \mathrm{T}, \mathsf{r} \rangle$ such that $\mathsf{r}(\varepsilon) = (q_0, \alpha_0)$ and for each $x \in \mathrm{T}$ with $\mathsf{r}(x) = (q, \gamma \cdot \alpha)$, there is a tuple $\langle (q_1, \beta 1), \ldots, (q_k, \beta_k) \rangle \in \delta(q, \mathsf{V}(x), \gamma)$ for some $k \in \mathbb{N}$, such that, for all $1 \leq i \leq k$, $\mathsf{r}(x \cdot i) = (q_i, \beta_i \cdot \alpha)$ if $\gamma \neq \perp$, and $\mathsf{r}(x \cdot i) = (q_i, \beta_i \cdot \perp)$, otherwise.

Given a path $\pi$ starting from $\varepsilon$, by $\mathsf{inf}_{\mathsf{r}}(\pi)$ we denote the subset of states $q$ such that there are infinitely many $x \in \pi$ such that $\mathsf{r}(x) \in \{q\} \times \Gamma_{\perp}^{*}$. A path satisfies a *parity* condition $\mathrm{F} = \{\mathrm{F}_1, \ldots, \mathrm{F}_k\}$, with $\mathrm{F}_i \subseteq \mathrm{F}_{i+1}$, for all $i < k$, and $\mathrm{F}_k = \mathrm{Q}$, if the minimum index $1 \leq i \leq k$ such that $\mathsf{inf}_{\mathsf{r}}(\pi) \cap \mathrm{F}_i \neq \emptyset$ is even. A run $\langle \mathrm{T}, \mathsf{r} \rangle$ is *accepting* if every path satisfies the acceptance condition. The PD-NTA $\mathcal{A}$ accepts an input tree $\langle \mathrm{T}, \mathsf{V} \rangle$ iff there is an accepting run of $\mathcal{A}$ over it. The language of $\mathcal{A}$, denoted by $\mathcal{L}(\mathcal{A})$, contains all the trees accepted by $\mathcal{A}$. The emptiness problem for PD-NTA is to decide, for a given $\mathcal{A}$, whether $\mathcal{L}(\mathcal{A}) = \emptyset$. In [KPV02] it is reported the following.

**Theorem 4.3.1** *The emptiness problem for a parity* PD-NTA *is* EXPTIME-COMPLETE.

In several branching-time temporal-logic verification settings, the automata-theoretic approach has been fruitfully applied. Very close to our case are the procedures deployed for model checking pushdown systems over CTL$^\star$ specifications [BEM97, BMP10, Boz06] and finite-state CGSs over ATL$^\star$ specifications [AHK02]. The former is a top-down procedure that first builds an automaton accepting all the trees that satisfy the formula and then checks for the membership problem of the tree unwinding of the pushdown model. Precisely, to get a tight complexity, it starts with a single-exponential alternating [1] parity tree automaton and the membership problem results in a special alternating pushdown tree automaton named *one-letter*, with no blow-up in size, whose emptiness can be checked in exponential-time[2] resulting in an overall doubly-exponential time solution. The procedure for ATL$^\star$, instead, uses a doubly-exponential bottom-up approach based on the idea of labeling each state of the structure with subformulas true in that state. In our setting we can neither proceed with the membership problem nor use a bottom-up procedure. Indeed, because of ATL$^\star$, we need to consider not just the unwinding of the model but the tree execution induced by the player existentially quantified in the formula. Moreover, because of the possible infinite number of configurations induced by the PGS, a bottom-up procedure could never terminate. For this reason, we use a top-down approach that constructs a doubly exponential PD-NTA that simultaneously checks whether a tree is an execution of the structure and a model of the formula. As far as we know, this is the first top-down automata-theoretic approach exploited for ATL$^\star$. Some details about this automata construction are reported in the following.

**Theorem 4.3.2** *The model-checking problem for* ATL$^\star$ *on* PGS *can be solved in 3*EXPTIME.

[1] Automata having as transition relation a positive Boolean combination of states and directions [KVW00].

[2] Recall that in general the emptiness check for alternating pushdown automata is undecidable [KPV02].

*Proof.* We give an intuition behind the automata construction by providing some details on how to extend the one introduced in [BEM97] used to solve the model-checking problem for branching-time specifications over pushdown systems. The mentioned approach starts with a tree automaton accepting all tree models of a formula $\varphi$, namely the formula automaton $\mathcal{A}_\varphi$, over which one can build a PD-NTA $\mathcal{A}_{\mathcal{P},\varphi}$ accepting the unwinding of the pushdown structure $\mathcal{P}$ iff it is contained in the language of the formula automaton $\mathcal{A}_\varphi$. To handle ATL$^\star$, one can start with a doubly-exponential parity tree automaton as an adaptation of the one provided in [Sch08][3]. Moreover, in order to correctly evaluate the formula over a PGS we need not just to consider the unwinding of the structure but rather the execution trees induced by the formula and precisely from the players existentially quantified in it. This results in selecting at each node subsets of children upon the choices of the players. As the number of these subsets is linear in the number of the decisions of the structure, the overall size of the PD-NTA we construct remains doubly-exponential. Thus, from Theorem 4.3.1 on the previous page we derive a 3EXPTIME procedure. □

### 4.3.2 A Lower-bound Complexity

In this section, the model-checking problem for ATL$^\star$ over PGSs is shown to be 2EXPSPACE-HARD by means of a reduction from the model-checking problem for *quantitative* ATL$^\star$ over *one-counter games* [Ves14]. We first give the definition of one-counter games.

**Definition 4.3.1 (One-counter Game)** *A* one-counter game *is a tuple* $\mathcal{M} = \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Loc}, (\mathrm{Loc}_a)_{a\in\mathrm{Ag}}, l_\mathrm{o}, R, \mathsf{L}\rangle$ *where* AP *is a finite set of atomic propositions,* Ag *is a finite set of agents,* Loc *is a finite set of locations,* $(\mathrm{Loc}_a)_{a\in\mathrm{Ag}}$ *is a partition of* Loc*, assigning to each agent its own set of locations,* $R \subseteq \mathrm{Loc} \times \{-1, 0, 1\} \times \mathrm{Loc}$ *is a transition relation,* $\mathsf{L} : \mathrm{St} \to 2^{\mathrm{AP}}$ *is a labeling function, and* $l_\mathrm{o}$ *is an initial location.*

Intuitively, a computation in $\mathcal{M}$ starts from the initial location $l_\mathrm{o}$ with a counter set to the value 0. Then, at each location $l \in \mathrm{Loc}_a$, the owner agent $a$ selects a successor $l'$ and updates the counter according to the transition relation $R$, *i.e.*, if $a$ selects the triple $(l, \iota, l') \in R$, then the computation moves to the location $l'$ and the current value $\mathtt{v}$ of the counter is updated to $\mathtt{v} + \iota$. As additional feature, if the counter value is 0, then the transitions providing a decreasing of the counter are disabled. Formally, a computation in $\mathcal{M}$ is an infinite sequence of pairs $\eta = (l_\mathrm{o}, v_\mathrm{o}) \cdot (l_1, v_1) \ldots \in (\mathrm{Loc} \times \mathbb{N})^\omega$ such that, for all $i \in \mathbb{N}$, we have that $(l_i, v_{i+1} - v_i, l_{i+1}) \in R$. For convenience, by $\mathsf{loc}(\eta) \in \mathrm{Loc}^\omega$ and $\mathsf{cnt}(\eta) \in \mathbb{N}^\omega$, we denote the projections of $\eta$ on locations and counter values, respectively.

One-counter games can be used to interpret *quantitative* ATL$^\star$ (QATL$^\star$, for short), which is obtained from ATL$^\star$, by extending the syntax as in the following.

**Definition 4.3.2 (Quantitative** ATL$^\star$**)** *Formulas of* QATL$^\star$ *are built inductively from the set of atomic propositions* AP *and agents* Ag*, by using the following grammar, where* $p \in \mathrm{AP}$*,* $r$ *is a variable ranging on* $\mathbb{N}$*,* $\bowtie \in \{\leq, <, =, >, \geq, \equiv_k\}$*,* $c \in \mathbb{N}$*, and* $\mathrm{A} \subseteq \mathrm{Ag}$*:*

[3] In [Sch08] it is given a single-exponential alternating automaton that can be easily translated into a non-deterministic one with a single exponential-time blowup.

$$\varphi := p \mid r \bowtie c \mid \neg\varphi \mid \varphi \wedge \varphi \mid \mathtt{X}\varphi \mid \varphi \mathtt{U} \varphi \mid \langle\langle \mathrm{A} \rangle\rangle \varphi.$$

As in ATL$^\star$, a QATL$^\star$ sentence is a Boolean combination of formulas of the form $\langle\langle \mathrm{A} \rangle\rangle \psi$. Formulas in QATL$^\star$ are interpreted over one-counter games and computations. For each one-counter game $\mathcal{M}$ and a computation $\eta$, and two natural numbers $\mathtt{c}, k \in \mathbb{N}$, we say that $\mathcal{M}, \eta \models r \bowtie c$ if $(\mathtt{cnt}(\eta))_0 \bowtie c$, for all $\bowtie \in \{\leq, <, =, >, \geq, \equiv_k\}$, while for the atomic proposition, Boolean, temporal, and agent quantification cases, the semantics is defined as in ATL$^\star$. Also in the case of QATL$^\star$ sentences, the satisfaction on a given computation $\eta$ depends only on its first configuration. Then, the relation $\mathcal{M}, (l, i) \models \varphi$ is well-defined and, so, the model-checking problem consists on checking whether $\mathcal{M}, (l_0, 0) \models \varphi$.

In [Ves14], it has been shown that the model-checking problem for QATL$^\star$ over one-counter games, with counter coded in unary, is 2ExpSpace-complete and PSpace-complete for QATL$^\star$ formulas of bounded size. We now show that such a problem can be reduced to the model-checking problem of ATL$^\star$ over PGSs. Indeed, consider the following construction.

**Construction 4.3.1** *Let $\mathcal{M} = \langle \mathrm{AP}, \mathrm{Ag}, \mathrm{Loc}, (\mathrm{Loc}_a)_{a \in \mathrm{Ag}}, s_0, R, \mathsf{L} \rangle$ be a one-counter model and $\varphi$ an* QATL$^\star$ *formula with $n_1, \ldots, n_k$ being the natural numbers for which there is an occurrence of $r \equiv_{n_i} c_i$ in $\varphi$, for some $c_i \in \mathbb{N}$. Then the derived* PGS *is given by $\mathcal{P}_{\mathcal{M},\varphi} = \langle \mathrm{AP}', \mathrm{Ag}', \mathrm{Ac}', \mathrm{Loc}', \Gamma, \mathsf{tr}, \mathsf{ap}, s_0 \rangle$ where: $\mathrm{AP}' = \mathrm{AP} \cup \{o, \mathtt{zero}, \sharp, \mathtt{mod}_{n_1}, \ldots, \mathtt{mod}_{n_k}, \mathtt{mod}^0_{n_1}, \ldots, \mathtt{mod}^{n_1-1}_{n_1}, \ldots, \mathtt{mod}^0_{n_k}, \ldots \mathtt{mod}^{n_k-1}_{n_k}\}$, $\mathrm{Ag}' = \mathrm{Ag} \cup \{\mathtt{cnt}\}$, $\mathrm{Ac}' = \mathrm{Loc} \times \{-1, 0, +1\} \cup \{o, \sharp, \mathtt{mod}_{n_1}, \ldots \mathtt{mod}_{n_k}\}$, $\mathrm{Loc}' = \mathrm{Loc} \cup \{l_\sharp, l^{n_1}_0, \ldots, l^{n_1}_{n_1-1}, \ldots, l^{n_k}_0, \ldots, l^{n_k}_{n_k-1}\}$, $\Gamma = \{1\}$, $\mathsf{ap}$ is such that: $\mathsf{ap}(l, \gamma) = \mathsf{L}(l) \cup \{o\}$, for all $l \in \mathrm{St}$, $\mathsf{ap}(l_\sharp, 1) = \{\sharp\}$ and $\mathsf{ap}(l_\sharp, \bot) = \{\sharp, \mathtt{zero}\}$, and $\mathsf{ap}(l^{n_i}_j, 1) = \{\mathtt{mod}_{n_i}\}$ and $\mathsf{ap}(l^{n_i}_j, \bot) = \{\mathtt{mod}_{n_i}, \mathtt{mod}^j_{n_i}\}$. The transition function is given as follows. For each $(l, \iota, l') \in R$, with $l \in \mathrm{Loc}_a$, we have that: $\mathsf{tr}(l, \gamma, \delta[\mathtt{cnt} \mapsto o][a \mapsto (l', \iota)]) = (l', \varepsilon)$ if $\iota = -1$, $\mathsf{tr}(l, \gamma, \delta[\mathtt{cnt} \mapsto o][a \mapsto (l', \iota)]) = (l', 1)$ if $\iota = 0$, $\mathsf{tr}(l, \gamma, \delta[\mathtt{cnt} \mapsto o][a \mapsto (l', \iota)]) = (l', 1 \cdot 1)$ if $\iota = +1$. $\mathsf{tr}(l, \gamma, \delta[\mathtt{cnt} \mapsto \sharp]) = (l_\sharp, \varepsilon)$, $\mathsf{tr}(l, \gamma, \delta[\mathtt{cnt} \mapsto \mathtt{mod}_{n_i}) = (l^{n_i}_1, \varepsilon)$, $\mathsf{tr}(l_\sharp, \gamma, \delta) = (l_\sharp, \varepsilon)$, $\mathsf{tr}(l^{n_j}_i, 1, \delta) = (l^{n_j}_{(i+1) \bmod n_j}, \varepsilon)$, $\mathsf{tr}(l^{n_j}_i, \bot, \delta) = (l^{n_j}_i, \varepsilon)$.*

Intuitively, the PGS $\mathcal{P}_{\mathcal{M},\varphi}$ emulates the counter in $\mathcal{M}$ with a one-symbol stack, in the usual way. When the transition in $\mathcal{M}$ asks for an increasing of the counter, then a the symbol 1 is pushed on the top of the stack. On the other hand, when the transition in $\mathcal{M}$ asks for a decreasing of the counter, then, the symbol in the top of the stack is popped out. Note that the original model never decreases the counter set to the value 0. This means that $\mathcal{P}_{\mathcal{M},\varphi}$ never performs a pop when the stack is empty. Finally, if the transition in $\mathcal{M}$ does not modify the counter, then $\mathcal{P}_{\mathcal{M},\varphi}$ does the same. For what the labeling regards, for a location $l \in \mathrm{Loc}$ we extend the labeling given in $\mathcal{M}$ with an auxiliary atomic proposition $o$, to keep track in the execution that $\mathcal{P}_{\mathcal{M},\varphi}$ is emulating the one generated in $\mathcal{M}$. For what the additional locations concerns, we label configurations having $l_\sharp$ as location with the atomic proposition $\sharp$, to keep track that the execution is performing a check on the size of the stack. Moreover, we label $(l, \bot)$ also with the atomic proposition $\mathtt{zero}$, in order to remember that the stack has been emptied. For what the locations of the form $l^{n_j}_i$ regards, we label all the corresponding configurations with the atomic proposition $\mathtt{mod}_{n_j}$ to remember that the

execution is performing a check modulo $n_j$ of the stack, and the one with the symbol $\perp$ also with $\mathtt{mod}_i^{n_j}$ to remember that the stack has been emptied in $m$ steps, with $m \equiv_{n_j} i$. Moreover, in order to emulate the checks of the QATL$^\star$ formula, we add some auxiliary location. For example, we add a location $l_\sharp$ from which the game loops indefinitely and performs a pop of the stack up to empty it, whatever is the decision made by the agents. Agent cnt has the power to enter the location $l_\sharp$, from each original location $l \in \mathrm{Loc}$, at any execution point.

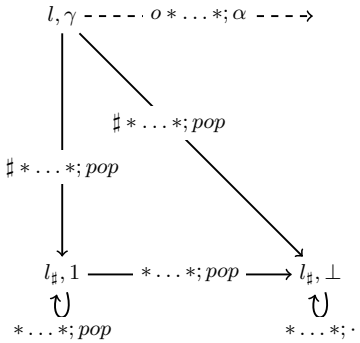



Figure 4.2: The transition to the counting location.

To better understand this point, assume the execution to be in a location $l$ of the PGS, as depicted in Figure 4.2. In the case the agent cnt picks the action $o$, then the execution moves to a location and makes a stack operation according to the choices of the rest of the agents, emulating the execution in $\mathcal{M}$. On the contrary, in the case the agent cnt picks the action $\sharp$, then the execution moves to the location $l_\sharp$. Observe that, once a transition to $l_\sharp$ is performed, the number of steps (including such transition) in which the stack gets empty is exactly the length of its content.

Thus, for each configuration $s = (l, \alpha)$ the fact that $|\alpha| \leq \mathtt{c}$, for a given $\mathtt{c} \in \mathbb{N}$, is equivalent to the fact that $\mathcal{P}, s \models \langle\!\langle \{\mathtt{cnt}\} \rangle\!\rangle (\mathtt{X}\sharp \wedge \mathtt{X}^{\mathtt{c}}\mathtt{zero})$ [4]. Indeed, with the temporal property $\mathtt{X}\sharp$ we are forcing the agent cnt to move on $l_\sharp$. Then, the stack is emptied in exactly $|\alpha|$ steps, so less than c, which implies that the temporal specification $\mathtt{X}^{\mathtt{c}}\mathtt{zero}$ is satisfied. From this, we can derive that $\mathcal{P}, s \models \langle\!\langle \mathtt{cnt} \rangle\!\rangle (\mathtt{X}\sharp \wedge \mathtt{X}^{\mathtt{c}}\mathtt{zero})$ iff $\mathcal{M}, (l, |\alpha|) \models \mathtt{r} \leq \mathtt{c}$. By exploiting the equivalences $(\mathtt{r} < \mathtt{c}) \leftrightarrow (\mathtt{r} \leq \mathtt{c} - 1)$, $(\mathtt{r} > \mathtt{c}) \leftrightarrow \neg(\mathtt{r} \leq \mathtt{c})$, $(\mathtt{r} \geq \mathtt{c}) \leftrightarrow \neg(\mathtt{r} > \mathtt{c})$, and $(\mathtt{r} = \mathtt{c}) \leftrightarrow ((\mathtt{r} \leq \mathtt{c}) \wedge (\mathtt{r} \geq \mathtt{c}))$, we can check any ordering of the counter by checking the satisfaction of a suitable ATL$^\star$ formula in $\mathcal{P}_{\mathcal{M},\varphi}$.

We can address the problem also for the formula $\mathtt{r} \equiv_n \mathtt{c}$. Indeed, we commit the agent cnt to pick a suitable action $\mathtt{mod}_n$ and so the PGS to enter a substructure suitably defined for this kind of check. As an example, consider the situation depicted in Figure 4.3 for $\equiv_2$. In this case, when the agent cnt pick its action $\mathtt{mod}_2$, then the execution moves to $l_2^1$ and, from that point onward, switches from $l_2^1$ to $l_2^0$ and viceversa, popping the top of the stack at each step. As soon as the stack is emptied, the execution loops in the location reached when the last popping has been performed. Also in this case, it is not hard to see that the execution starts looping on the location $l_2^i$ such that $i = |\alpha| \mathtt{mod} 2$. This implies that, for each configuration $s = (l, \alpha)$, the fact that $|\alpha| \equiv_n \mathtt{c}$, for some $n, \mathtt{c} \in \mathbb{N}$, is equivalent to check whether $\mathcal{P}_{\mathcal{M},\varphi}, (l, \alpha) \models \langle\!\langle \{\mathtt{cnt}\} \rangle\!\rangle (\mathtt{X}\mathtt{mod}_n \wedge \mathtt{F}\mathtt{mod}_n^{\mathtt{c}})$. Indeed, the temporal specification $\mathtt{X}\mathtt{mod}_n$ is used to force the agent cnt to pick the action $\mathtt{mod}_n$, while the formula

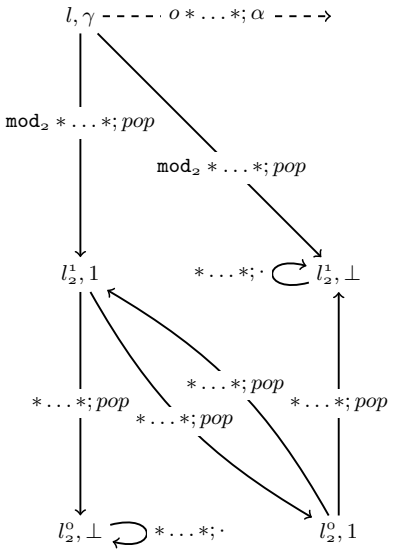



Figure 4.3: The transition to the counting modulo 2 location.

[4] By $\mathtt{X}^{\mathtt{c}}$ we are denoting c occurrences of the operator X.

$\texttt{Fmod}_n^{\texttt{c}}$ is satisfied iff the stack is emptied in a number $m$
of steps such that $m \equiv_n \texttt{c}$.

Finally, consider a QATL$^\star$ formula of the form $\varphi = \langle\langle \mathrm{A} \rangle\rangle \psi$. Interpreting $\varphi$ over a one-counter game $\mathcal{M}$ is equivalent to interpret the formula $\langle\langle \mathrm{A} \cup \{\texttt{cnt}\} \rangle\rangle \texttt{XG}o \wedge \psi$ on the derived PGS $\mathcal{P}$. Indeed, with the temporal specification $\texttt{XG}o$ we are essentially forcing the agent $\texttt{cnt}$ to always pick its action $o$. This implies that the other agents in $\mathrm{A}$ can apply the same strategies they use in $\mathcal{M}$ in order to satisfy the formula $\psi$ also in $\mathcal{P}$.

By applying the above reasoning, for a QATL$^\star$ formula $\varphi$ we define recursively the following transformation $\tau(\varphi)$: $\tau(\texttt{p}) = \texttt{p}$; $\tau(\texttt{r} \leq \texttt{c}) = \langle\langle \{\texttt{cnt}\} \rangle\rangle (\texttt{X}\sharp \wedge \texttt{X}^{\texttt{c}}\texttt{zero})$; $\tau(\texttt{r} \equiv_n \texttt{c}) = \langle\langle \{\texttt{cnt}\} \rangle\rangle (\texttt{Xmod}_n \wedge \texttt{Fmod}_n^{\texttt{c}})$; $\tau(\neg\psi) = \neg\tau(\psi)$; $\tau(\psi_1 \wedge \psi_2) = \tau(\psi_1) \wedge \tau(\psi_2)$; $\tau(\texttt{X}\psi) = \texttt{X}\tau(\psi)$; $\tau(\psi_1 \texttt{U} \psi_2) = \tau(\psi_1) \texttt{U} \tau(\psi_2)$; $\tau(\langle\langle \mathrm{A} \rangle\rangle \psi) = \langle\langle \mathrm{A} \cup \{\texttt{cnt}\} \rangle\rangle (\texttt{XG}o \wedge \tau(\psi))$.

It is not hard to prove the following theorem.

**Theorem 4.3.3 (Model-checking Reduction)** *For a given one-counter game $\mathcal{M}$ and a* QATL$^\star$ *formula $\varphi$, it holds that $\mathcal{M} \models \varphi$ iff $\mathcal{P}_{\mathcal{M},\varphi} \models \tau(\varphi)$*

Note that, for counters coded in unary, we have that the construction of the PGS reported above is of size polynomial in the size of the one-counter game. From this and from Theorem 4.3.3 we derive the following.

**Corollary 4.3.1 (Model-checking Lower-bound)** *The model-checking problem for* ATL$^\star$ *over* PGS *is in* 2EXPSPACE-HARD.

# Bibliography


[ABE+05] R. Alur, M. Benedikt, K. Etessami, P. Godefroid, T. W. Reps, and M. Yannakakis. Analysis of Recursive State Machines. *ACM Trans. Program. Lang. Syst.*, 27(4):786–818, 2005.

[ABK13] S. Almagor, U. Boker, and O. Kupferman. Formalizing and Reasoning about Quality. In *ICALP'13*, volume 7966 of *LNCS*, pages 15–27, 2013.

[ABK14] S. Almagor, U. Boker, and O. Kupferman. Discounting in LTL. In *TACAS'14*, volume 8413 of *LNCS*, pages 424–439. Springer, 2014.

[AF84] M. Aigner and M. Fromme. A Game of Cops and Robbers. *Discrete Applied Mathematics*, 8(1):1 – 12, 1984.

[ÅGJ07] T. Ågotnes, V. Goranko, and W. Jamroga. Alternating-Time Temporal Logics with Irrevocable Strategies. In *Theoretical Aspects of Rationality and Knowledge'07*, pages 15–24, 2007.

[AHK02] R. Alur, T.A. Henzinger, and O. Kupferman. Alternating-Time Temporal Logic. *Journal of the ACM*, 49(5):672–713, 2002.

[ALM+13] B. Aminof, A. Legay, A. Murano, O. Serre, and M. Y. Vardi. Pushdown Module Checking with Imperfect Information. *Inf. Comput.*, 223:1–17, 2013.

[BCHJ09] R. Bloem, K. Chatterjee, T. Henzinger, and B. Jobstmann. Better Quality in Synthesis through Quantitative Objectives. In *CAV'09*, volume 5643 of *LNCS*, pages 140–156. Springer, 2009.

[Bel14] Francesco Belardinelli. Reasoning about Knowledge and Strategies: Epistemic Strategy Logic. In *SR*, pages 27–33, 2014.

[BEM97] A. Bouajjani, J. Esparza, and O. Maler. Reachability Analysis of Pushdown Automata: Application to Model-Checking. In *CONCUR'97*, pages 135–150, 1997.

[BJ14] N. Bulling and W. Jamroga. Comparing variants of strategic ability: how uncertainty and memory influence general properties of games. *Autonomous Agents and Multi-Agent Systems*, 28(3):474–518, 2014.

[BJD08] Nils Bulling, Wojciech Jamroga, and Jürgen Dix. Reasoning about temporal properties of rational play. *Ann. Math. Artif. Intell.*, 53(1-4):51–114, 2008.

[BLLM09] T. Brihaye, A. Da Costa Lopes, F. Laroussinie, and N. Markey. ATL with Strategy Contexts and Bounded Memory. In *Logical Foundations of Computer Science'09*, LNCS 5407, pages 92–106. Springer, 2009.

[BMP10] L. Bozzelli, A. Murano, and A. Peron. Pushdown Module Checking. *Formal Methods in System Design*, 36(1):65–95, 2010.

[Boz06] L. Bozzelli. Complexity Results on Branching-Time Pushdown Model Checking. In *VMCAI'06*, pages 65–79, 2006.

[Büc62] J.R. Büchi. On a Decision Method in Restricted Second-Order Arithmetic. In *International Congress on Logic, Methodology, and Philosophy of Science'62*, pages 1–11. Stanford University Press, 1962.

[Bul14] N. Bulling. A Survey of Multi-Agent Decision Making. *KI*, 28(3):147–158, 2014.

[CBC13] C. Chareton, J. Brunel, and D. Chemouil. Towards an Updatable Strategy Logic. In *Proceedings 1st International Workshop on Strategic Reasoning, SR 2013, Rome, Italy, March 16-17, 2013.*, pages 91–98, 2013.

[CE81] E.M. Clarke and E.A. Emerson. Design and Synthesis of Synchronization Skeletons Using Branching-Time Temporal Logic. In *Logic of Programs'81*, LNCS 131, pages 52–71. Springer, 1981.

[CGP02] E.M. Clarke, O. Grumberg, and D.A. Peled. *Model Checking*. MIT Press, 2002.

[CHJ08] K. Chatterjee, T. Henzinger, and B. Jobstmann. Environment Assumptions for Synthesis. In *CONCUR'08*, volume 5201 of *LNCS*, pages 147–161. Springer, 2008.

[CHP07] K. Chatterjee, T.A. Henzinger, and N. Piterman. Strategy Logic. In *Concurrency Theory'07*, LNCS 4703, pages 59–73. Springer, 2007.

[Chu63] A. Church. Logic, Arithmetics, and Automata. In *International Congress of Mathematicians'62*, pages 23–35, 1963.

[ČLMM14] P. Čermák, A. Lomuscio, F. Mogavero, and A. Murano. MCMAS-SLK: A Model Checker for the Verification of Strategy Logic Specifications. In *CAV'14*, LNCS 8559, pages 524–531. Springer, 2014.

[CMJ04] K. Chatterjee, R. Majumdar, and M. Jurdzinski. On Nash Equilibria in Stochastic Games. In *CSL'04*, volume 3210 of *LNCS*, pages 26–40. Springer, 2004.

[CY98] C. Courcoubetis and M. Yannakakis. Markov Decision Processes and Regular Events. *IEEE Transaction on automatic control*, 43(10):1399 – 1418, 1998.

[DLM10] A. Da Costa, F. Laroussinie, and N. Markey. ATL with Strategy Contexts: Expressiveness and Model Checking. In *Foundations of Software Technology and Theoretical Computer Science'10*, LIPIcs 8, pages 120–132, 2010.

[DLM12] A. Da Costa, F. Laroussinie, and N. Markey. Quantified CTL: Expressiveness and model checking - (extended abstract). In *CONCUR*, volume 7454 of *Lecture Notes in Computer Science*, pages 177–192. Springer, 2012.

[EF95] H.D. Ebbinghaus and J. Flum. *Finite Model Theory.* Perspectives in Mathematical Logic. Springer, 1995.

[EH86] E.A. Emerson and J.Y. Halpern. "Sometimes" and "Not Never" Revisited: On Branching Versus Linear Time. *Journal of the ACM*, 33(1):151–178, 1986.

[FKL10] D. Fisman, O. Kupferman, and Y. Lustig. Rational Synthesis. In *TACAS*, pages 190–204, 2010.

[FMP08] A. Ferrante, A. Murano, and M. Parente. Enriched $\mu$-Calculi Module Checking. *Logical Methods in Computer Science*, 4(3), 2008.

[FS10] B. Finkbeiner and S. Schewe. Coordination Logic. In *Computer Science Logic'10*, LNCS 6247, pages 305–319. Springer, 2010.

[GTW02] E. Grädel, W. Thomas, and T. Wilke. *Automata, Logics, and Infinite Games: A Guide to Current Research.* LNCS 2500. Springer, 2002.

[Har84] D. Harel. A Simple Highly Undecidable Domino Problem. In *Logic and Computation Conference'84*. North-Holland, 1984.

[Hen10] T.A. Henzinger. From Boolean to Quantitative Notions of Correctness. In *POPL'10*, pages 157–158. ACM, 2010.

[HLW13] A. Herzig, E. Lorini, and D. Walther. Reasoning about Actions Meets Strategic Logics. In *Logic, Rationality, and Interaction*, pages 162–175. Springer, 2013.

[Hod93] W. Hodges. *Model Theory.* Encyclopedia of Mathematics and its Applications. Cambridge, 1993.

[Hod01] W. Hodges. *Elementary Predicate Logic.*, volume 1 of *Handbook of philosophical logic*. Springer, 2001.

[HSW13] C. Huang, S. Schewe, and F. Wang. Model-Checking Iterated Games. In *Tools and Algorithms for the Construction and Analysis of Systems*, pages 154–168. Springer, 2013.

[JM14] W. Jamroga and A. Murano. On module checking and strategies. In *AAMAS'14*, pages 701–708, 2014.

[JvdH04] W. Jamroga and W. van der Hoek. Agents that Know How to Play. *Fundamenta Informaticae*, 63(2-3):185–219, 2004.

[Kel76] R.M. Keller. Formal Verification of Parallel Programs. *Communication of the ACM*, 19(7):371–384, 1976.

[Kol85] F. Kolaitis. Game Quantification. In J Barwise and S. Feferman, editors, *Handbook of Model-Theoretic Logics*, pages 365–421. SpringerVerlag, 1985.

[Koz83] D. Kozen. Results on the Propositional muCalculus. *Theoretical Computer Science*, 27(3):333–354, 1983.

[KPV02] O. Kupferman, N. Piterman, and M. Y. Vardi. Pushdown Specifications. In *LPAR'02*, pages 262–277, 2002.

[KPV14] O. Kupferman, G. Perelli, and M. Y. Vardi. Synthesis with Rational Environments. In *EUMAS'14*, 2014. To appear.

[Kri63] S.A. Kripke. Semantical Considerations on Modal Logic. *Acta Philosophica Fennica*, 16:83–94, 1963.

[KV98] O. Kupferman and M.Y. Vardi. Weak Alternating Automata and Tree Automata Emptiness. In *Symposium on Theory of Computing'98*, pages 224–233, 1998.

[KV99] O. Kupferman and M. Y. Vardi. Church's problem revisited. *Bulletin of Symbolic Logic*, 5(2):245–263, 1999.

[KVW00] O. Kupferman, M.Y. Vardi, and P. Wolper. An Automata Theoretic Approach to Branching-Time Model Checking. *Journal of the ACM*, 47(2):312–360, 2000.

[KVW01] O. Kupferman, M.Y. Vardi, and P. Wolper. Module Checking. *Information and Computation*, 164(2):322–344, 2001.

[LM13] F. Laroussinie and N. Markey. Satisfiability of ATL with Strategy Contexts. In *GandALF*, volume 119 of *EPTCS*, pages 208–223, 2013.

[LR06] A. Lomuscio and F. Raimondi. MCMAS: A Model Checker for Multi-agent Systems. In *Tools and Algorithms for the Construction and Analysis of Systems'06*, LNCS 3920, pages 450–454. Springer, 2006.

[Mar75] A.D. Martin. Borel Determinacy. *Annals of Mathematics*, 102(2):363–371, 1975.

[MMPV11] F. Mogavero, A. Murano, G. Perelli, and M.Y. Vardi. Reasoning About Strategies: On the Model-Checking Problem. Technical report, arXiv, 2011.

[MMPV12] F. Mogavero, A. Murano, G. Perelli, and M.Y. Vardi. What Makes ATL$^\star$ Decidable? A Decidable Fragment of Strategy Logic. In *Concurrency Theory'12*, LNCS 7454, pages 193–208. Springer, 2012.

[MMPV14] F. Mogavero, A. Murano, G. Perelli, and M. Y. Vardi. Reasoning About Strategies: On the Model-Checking Problem. *ACM Trans. Comput. Log.*, 15(4):34, 2014.

[MMS13] F. Mogavero, A. Murano, and L. Sauro. On the Boundary of Behavioral Strategies. In *Logic in Computer Science'13*, pages 263–272. IEEE Computer Society, 2013.

[MMS14] F. Mogavero, A. Murano, and L. Sauro. A Behavioral Hierarchy of Strategy Logic. In *CLIMA XV*, LNCS 8624, pages 869–876. Springer, 2014.

[MMV10a] F. Mogavero, A. Murano, and M.Y. Vardi. Reasoning About Strategies. In *Foundations of Software Technology and Theoretical Computer Science'10*, LIPIcs 8, pages 133–144. Leibniz-Zentrum fuer Informatik, 2010.

[MMV10b] F. Mogavero, A. Murano, and M.Y. Vardi. Relentful Strategic Reasoning in Alternating-Time Temporal Logic. In *Logic for Programming Artificial Intelligence and Reasoning'10*, LNAI 6355, pages 371–387. Springer, 2010.

[MP14] F. Mogavero and G. Perelli. On the Remarkable Features of Binding Forms. *CoRR*, abs/1404.1531, 2014.

[MS85] D. E. Muller and P. E. Schupp. The Theory of Ends, Pushdown Automata, and Second-Order Logic. *Theor. Comput. Sci.*, 37:51–75, 1985.

[MS87] D.E. Muller and P.E. Schupp. Alternating Automata on Infinite Trees. *Theoretical Computer Science*, 54(2-3):267–276, 1987.

[MS95] 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. *Theoretical Computer Science*, 141(1-2):69–107, 1995.

[MSS88] D.E. Muller, A. Saoudi, and P.E. Schupp. Weak Alternating Automata Give a Simple Explanation of Why Most Temporal and Dynamic Logics are Decidable in Exponential Time. In *Logic in Computer Science'88*, pages 422–427. IEEE Computer Society, 1988.

[Mye97] R.B. Myerson. *Game Theory: Analysis of Conflict*. Harvard University Press, 1997.

[NR01] N. Nisan and A. Ronen. Algorithmic Mechanism Design. *Games and Economic Behavior*, 35(1-2):166–196, 2001.

[NRTV07] N. Nisan, T. Roughgarden, E. Tardos, and V.V. Vazirani. *Algorithmic Game Theory*. Cambridge University Press, 2007.

[OR94] M.J. Osborne and A. Rubinstein. *A Course in Game Theory.* MIT Press, 1994.

[Pau02] M. Pauly. A Modal Logic for Coalitional Power in Games. *Journal of Logic and Computation*, 12(1):149–166, 2002.

[Pin07] S. Pinchinat. A Generic Constructive Solution for Concurrent Games with Expressive Constraints on Strategies. In *Automated Technology for Verification and Analysis'07*, LNCS 4762, pages 253–267. Springer, 2007.

[Pnu77] A. Pnueli. The Temporal Logic of Programs. In *Foundation of Computer Science'77*, pages 46–57. IEEE Computer Society, 1977.

[PP04] D. Perrin and J. Pin. *Infinite Words.* Pure and Applied Mathematics. Elsevier, 2004.

[PR89] A. Pnueli and R. Rosner. On the synthesis of a reactive module. In *POPL'89*, pages 179–190, 1989.

[QS81] J.P. Queille and J. Sifakis. Specification and Verification of Concurrent Programs in Cesar. In *Symposium on Programming'81*, LNCS 137, pages 337–351. Springer, 1981.

[Rab69] M.O. Rabin. Decidability of Second-Order Theories and Automata on Infinite Trees. *Transactions of the American Mathematical Society*, 141:1–35, 1969.

[Sch08] S. Schewe. ATL* Satisfiability is 2ExpTime-Complete. In *International Colloquium on Automata, Languages, and Programming'08*, LNCS 5126, pages 373–385. Springer, 2008.

[Sel75] R. Selten. Reexamination of the Perfectness Concept for Equilibrium Points in Extensive Games. *International Journal of Game Theory*, 4(1):25–55, 1975.

[Sis83] A.P. Sistla. *Theoretical Issues in the Design and Cerification of Distributed Systems.* PhD thesis, Harvard University, Cambridge, MA, USA, 1983.

[SVW87] A.P. Sistla, M.Y. Vardi, and P. Wolper. The Complementation Problem for Büchi Automata with Applications to Temporal Logic. *Theoretical Computer Science*, 49:217–237, 1987.

[Tho90] W. Thomas. Automata on Infinite Objects. In *Handbook of Theoretical Computer Science (vol. B)*, pages 133–191. MIT Press, 1990.

[Var88] M.Y. Vardi. A Temporal Fixpoint Calculus. In *Principles of Programming Languages'88*, pages 250–259. Association for Computing Machinery, 1988.

[Var96] M.Y. Vardi. Why is Modal Logic So Robustly Decidable? In *Descriptive Complexity and Finite Models'96*, pages 149–184. American Mathematical Society, 1996.

[Ves14] Steen Vester. Model-checking Quantitative Alternating-time Temporal Logic on One-counter Game Models. Technical report, ArXiv, 2014.

[VS85] M.Y. Vardi and L.J. Stockmeyer. Improved Upper and Lower Bounds for Modal Logics of Programs: Preliminary Report. In *Symposium on Theory of Computing'85*, pages 240–251, 1985.

[VW86] M.Y. Vardi and P. Wolper. An Automata-Theoretic Approach to Automatic Program Verification. In *Logic in Computer Science'86*, pages 332–344. IEEE Computer Society, 1986.

[Wal96] I. Walukiewicz. Pushdown Processes: Games and Model Checking. In *CAV'96*, pages 62–74, 1996.

[Wal00] I. Walukiewicz. Model Checking CTL Properties of Pushdown Systems. In *FSTTCS'00*, pages 127–138, 2000.

[Wan61] H. Wang. Proving Theorems by Pattern Recognition II. *Bell System Technical Journal*, 40:1–41, 1961.

[WHY11] F. Wang, C. Huang, and F. Yu. A Temporal Logic for the Interaction of Strategies. In *CONCUR 2011*, pages 466–481. Springer, 2011.

[WvdHW07] Dirk Walther, Wiebe van der Hoek, and Michael Wooldridge. Alternating-Time Temporal Logic with Explicit Strategies. In *TARK*, pages 269–278, 2007.