Information and Computation 294 (2023) 105081

Contents lists available at ScienceDirect

Information and Computation

journal homepage: www.elsevier.com/locate/yinco

Taming Strategy Logic: Non-Recurrent Fragments

Massimo Benerecetti, Fabio Mogavero *, Adriano Peron

ARTICLE INFO

Article history:
Received 3 April 2023
Received in revised form 6 August 2023
Accepted 15 August 2023
Available online 24 August 2023

ABSTRACT

Strategy Logic (SL for short) is one of the prominent languages for reasoning about the strategic abilities of agents in a multi-agent setting. This logic extends LTL with first-order quantifiers over the agent strategies and encompasses other formalisms, such as ATL* and CTL*. The *model-checking problem* for SL and several of its fragments has been extensively studied. On the other hand, the picture is much less clear on the satisfiability front, where the problem is undecidable for the full logic. In this work, we study two fragments of *One-Goal* SL, where the nesting of sentences within temporal operators is constrained. We show that the *satisfiability problem* for these two logics, and for the corresponding fragments of ATL* and CTL*, is in ExpSpace and PSpace-complete, respectively.

1. Introduction

Temporal logic in its many flavours has proved to be a valuable tool for expressing and analysing properties of *reactive systems*. Different variants of the logic have been studied extensively in the past forty-five years, since the introduction of LTL by Pnueli [1]. The various *temporal languages* proposed, whether based on the *linear-time* model [2–6] or on the *branching-time* one [7–15], have been successfully applied to the formal verification of *closed nondeterministic systems*, namely systems where the nondeterministic choices are resolved within, *i.e.*, are under direct control of, the system itself. Systems that interact with an unknown environment, however, such as *system modules* or *multi-agent systems*, typically involve two forms of nondeterministic choices, those controlled by the system and those controlled by the external environment. This kind of systems, often referred to as *open systems*, connects to the notion of *alternating computation* [16,17] and naturally leads to a *game-theoretic interpretation*, where the system and the environment play a game against each other, by making the choices under their own control, in order to satisfy (*resp.*, falsify) some goal. Verifying a desired property for an open system then corresponds to checking for the existence of a strategy that the system can follow in order to ensure that its computations satisfy that property, regardless of the possible behaviours of (*i.e.*, the strategies followed by) the external environment.

A number of extensions of temporal logics specifically tailored to reason about open multi-agent systems and incorporating, implicitly or explicitly, the notion of strategy as a central element, have been proposed in the literature that can also express interesting game-theoretic notions, such as various forms of equilibria in games [18–23]. *Alternating-Time Temporal Logic* (ATL* , for short) was originally introduced by Alur, Henzinger, and Kupferman [24] and allows for reasoning about *strategic behaviour* of agents with temporal goals. This logic generalises the branching-time temporal logic CTL* [10,13] by replacing the path quantifiers, *there exists* "E" and *for all* "A", with *strategic modalities* of the form "⟨⟨A⟩⟩" and "[[A]]", for a set A of agents. These modalities can express cooperation and competition among the agents involved towards achieving some required temporal goals. In particular, they allow for selective quantifications over the paths resulting from an infinite game between a coalition of agents and its adversary, the complement coalition. *Strategy Logic* (SL, for short) [25–29],

* Corresponding author.
E-mail address: fabio.mogavero@unina.it (F. Mogavero).

https://doi.org/10.1016/j.ic.2023.105081

instead, extends LTL with two *strategy quantifiers*, the existential $\exists x$ and the universal $\forall x$, as well as agent bindings (a, x), where a is an agent and x a strategy variable. Intuitively, these elements can be read, respectively, as *"there exists a strategy x"*, *"for all strategies x"*, and *"bind agent a to the strategy associated with x"*. SL considers strategies as first class citizens and can express properties requiring an arbitrary alternation of the strategic quantifiers, as opposed to, *e.g.*, ATL*, which only allows for at most one such alternation. From a semantic viewpoint, this entails that SL can encode arbitrary functional dependencies among strategies, which may be crucial to express relevant multi-agent systems and non-trivial game-theoretic notions [28,29].

The *model-checking problem* for SL and for many of its fragments has been studied with some depth and is relatively well-understood [28,30–33]. The picture is, however, much less clear when *satisfiability* is considered. The full logic SL is known to be undecidable [26] on concurrent game structures, while it is decidable on turn-based ones [34,35]. The *one-goal* fragment (SL[1G], for short), where only a single binding prefix is allowed in any sentence (*i.e.*, a formula with no free variables), is decidable in 2ExpTime [36]. On the other hand, the *Boolean-Goal* fragment, which allows for Boolean combinations of bindings within a sentence but no nesting of bindings, is already undecidable [29]. Recently, the *flat* fragment of *conjunctive-goal* SL has been studied in [37], proving that the problem is in PSpace-complete and witnessing the rare phenomenon of a language with a satisfiability problem easier than the corresponding model-checking one, which remains 2ExpTime-complete. Such fragment allows for conjunctions of bindings but no nesting of temporal operators within a sentence.

In this work, we widen the picture, by studying larger non-flat fragments of SL[1G]. Specifically, we allow some forms of nesting of temporal operators, but prevent sentences in the first (*resp.*, second) argument of an until (*resp.*, release) operator. Essentially, temporal operators cannot reiterate the request of satisfaction of a sentence arbitrarily many times. The resulting fragment is, thus, called *non-recurrent* SL[1G] (SL$^{\not\bigcirc}$[1G], for short). We show that the fragment where the first (*resp.*, second) argument of an until (*resp.*, release) is restricted to a pure LTL formula can be decided in ExpSpace. If we further restrict those arguments to Boolean formulae, instead, we obtain a weaker fragment (WSL$^{\not\bigcirc}$[1G], for short) with a PSpace-complete decision problem. To prove these results, we first introduce a normal form for the models of satisfiable sentences of these fragments. The distinctive property of such models is that, along any of their paths, the number of branching points is linear in the length of the formula. To do that, a sentence is converted into a "skeleton", where it is split into layers at the beginning of each block of strategy quantifiers, and then skolemised to obtain a set of purely universally-quantified formulas in order to apply techniques from first-order logic [38,39]. Then, we introduce a novel class of tree automata, called *bounded-fork automata*, accepting trees with a bounded number of nodes with more than one successor along each path. We show that the emptiness problem for these automata, unlike for classic tree automata, can be decided in LogSpace. These results are key to obtaining the complexity bounds. Indeed, we can show that for any sentence φ of the two considered fragments, we can build a bounded-fork automaton of size doubly-exponential (*resp.*, singly-exponential) in the length of φ, accepting all and only its normal models. The ExpSpace and PSpace upper bounds for satisfiability, then, immediately follow from the complexity of the emptiness problem. The results also trickle down to suitable fragments of sublogics of SL such as ATL, ATL*, CTL, and CTL*.

Restrictions similar in vein to the non-recurrent one we study here have been considered in the past for LTL, CTL, and CTL*. In [40] the author introduces *flat*LTL, *flat*CTL, and *flat*CTL*, as fragments of the corresponding temporal logics where the next operator is not allowed and the first argument of both the until and the release operators can only accommodate propositional formulae. In their LTL form, these restrictions have been applied in several contexts, such as temporal logics enriched with constraints over data [41,42], analysis of discrete pushdown timed systems [43], and for the synthesis of hybrid systems [44]. In particular, the LTL fragment considered in [41,43] is a sublogic of the linear-time logic underlying WSL$^{\not\bigcirc}$[1G], while the one originally considered in [40] is not comparable to ours, as it restricts the first and not the second argument of the release operator, therefore still allowing for recurrent sentences. While both model-checking and satisfiability problems for flatLTL have been shown to be PSpace-complete [45,46], to the best of our knowledge, only expressiveness properties have been studied for flatCTL and flatCTL* .

The paper is structured as follows. In Section 2 we fix some standard notations and concepts useful in the remaining of the paper. In Section 3 after recalling SL[1G], the decidable fragment of SL, we provide the definition of SL$^{\not\bigcirc}$[1G]. In Section 4 we introduce and discuss normal models for SL$^{\not\bigcirc}$[1G]. In Section 5 we introduce the novel class of *bounded-fork tree automata* and *prefix-deterministic word automata* exploited in the following Section 6 where efficient satisfiability-checking algorithms are devised and complexity results are stated for the non-recurrent fragments of SL[1G]. Eventually, we conclude the paper with a discussion and some line of future investigation in Section 7.

2. Preliminaries

Games A *concurrent game structure* (CGS, for short) *w.r.t.* non-empty finite sets of *atomic propositions* AP and *agents* Ag is a tuple $\mathfrak{G} \triangleq \langle \mathrm{Ac}, \mathrm{Ps}, \tau, v_I, \lambda \rangle$, where Ac and Ps are countable non-empty sets of *actions* and *positions*, $v_I \in \mathrm{Ps}$ is an *initial position*, and $\lambda \colon \mathrm{Ps} \to 2^{\mathrm{AP}}$ is a *labelling function* mapping every position $v \in \mathrm{Ps}$ to the set of atomic propositions $\lambda(v) \subseteq \mathrm{AP}$ true at that position. A *decision* $d \in \mathrm{Dc} \triangleq \mathrm{Ac}^{\mathrm{Ag}}$ is a function that chooses an action for each agent. A *move function* $\tau \colon \mathrm{Ps} \times \mathrm{Dc} \to \mathrm{Ps}$ maps every position $v \in \mathrm{Ps}$ and decision $d \in \mathrm{Dc}$ to a position $\tau(v, d) \in \mathrm{Ps}$. By abuse of notation, $\tau \subseteq \mathrm{Ps} \times \mathrm{Ps}$ also denotes the *transition relation* between positions such that $(v, w) \in \tau$ *iff* $\tau(v, d) = w$ for some $d \in \mathrm{Dc}$. As usual, τ^+ (*resp.*, τ^*) is the transitive (*resp.*, reflexive and transitive) closure of τ.

A *path* $\pi \in \mathrm{Pth} \subseteq \mathrm{Ps}^\infty$ is a finite or infinite sequence of positions compatible with the move function, *i.e.*, $((\pi)_i, (\pi)_{i+1}) \in \tau$, for each $i \in [0, |\pi| - 1)$, and $\mathsf{fst}(\pi)$ is the first position of the sequence. The set $\mathrm{Pth}(v) \triangleq \{\pi \in \mathrm{Pth} \mid |\pi| > 0, \mathsf{fst}(\pi) = v\}$ denotes the set of paths starting at a position v. A *history* at v is a non-empty finite path $\rho \in \mathrm{Hst}(v) \triangleq \mathrm{Pth}(v) \cap \mathrm{Ps}^+$ starting at that position. Similarly, a *play* $\pi \in \mathrm{Play}(v) \triangleq \mathrm{Pth}(v) \cap \mathrm{Ps}^\omega$ at v is an infinite path starting at v.

A *strategy* rooted at v is a function $\sigma \in \mathrm{Str}(v) \triangleq \mathrm{Hst}(v) \to \mathrm{Ac}$ mapping histories to actions. A v-rooted *profile* $\xi \in \mathrm{Prf}(v) \triangleq \mathrm{Ag} \to \mathrm{Str}(v)$ associates agents with strategies. A path $\pi \in \mathrm{Pth}(v)$ is *compatible* with a v-rooted profile $\xi \in \mathrm{Prf}(v)$ if, for each $i \in [0, |\pi| - 1)$, it holds that $(\pi)_{i+1} = \tau((\pi)_i, d)$, for the unique decision $d \in \mathrm{Dc}$ such that $d(a) = \xi(a)((\pi)_{\leq i})$, for all agents $a \in \mathrm{Ag}$. The unique play induced by a profile ξ from position v is denoted by $\mathsf{play}(\xi, v)$.

A CGS $\mathfrak{G}$ is a *tree* if, for some set X:

1) Ps is a prefix-closed set of words in X^*,
2) $v_I = \varepsilon$ is the empty word, and
3) $(v, w) \in \tau$ *iff* $w = v \cdot x$ for some $x \in \mathrm{X}$, for all positions $v, w \in \mathrm{Ps}$.

As usual, $\tau^{-1} \colon \mathrm{Ps} \setminus \varepsilon \to \mathrm{Ps}$ denotes the *predecessor* function $\tau^{-1}(v \cdot x) \triangleq v$, for all $v \cdot x \in \mathrm{Ps} \setminus \varepsilon$ with $x \in \mathrm{X}$. Finally, a tree CGS $\mathfrak{G}$ is *k-fork*, for some $k \in \mathbb{N}$, if along every path $\pi \in \mathrm{Pth}(v_I)$ there are at most k forks, namely, $\left|\left\{i \in \mathbb{N} \mid |\tau((\pi)_i)| > 1\right\}\right| \leq k$.

Automata A *deterministic* (*resp.*, *nondeterministic*) *word automaton* (DWA (*resp.*, NWA), for short) is a tuple $\langle \Sigma, \mathrm{Q}, \delta, q_I, \mathrm{Q_F} \rangle$, where

- Σ and Q are the non-empty finite sets of *input symbols* and *states*;
- $q_I \in \mathrm{Q}$ is the *initial state*;
- $\mathrm{Q_F} \subseteq \mathrm{Q}$ is the subset of *final states*;
- $\delta \colon \mathrm{Q} \times \Sigma \to \mathrm{Q} \cup \{\bot, \top\}$ (*resp.*, $\delta \colon \mathrm{Q} \times \Sigma \to 2^{\mathrm{Q}}$) is the *deterministic* (*resp.*, *nondeterministic*) *transition function* mapping each state $q \in \mathrm{Q}$ and input symbol $\sigma \in \Sigma$ to the successor state (*resp.*, set of successor states) $\delta(q, \sigma)$, and $\bot$ and $\top$ are two implicit distinguished rejecting and accepting states used to simplify the constructions of this work (these implicit states are not needed in the case of nondeterministic automata).

By $\delta^* \colon \mathrm{Q} \times \Sigma^* \to \mathrm{Q} \cup \{\bot, \top\}$ we denote the lift of a deterministic transition function $\delta \colon \mathrm{Q} \times \Sigma \to \mathrm{Q} \cup \{\bot, \top\}$ from single symbols to words.

A *deterministic* (*resp.*, *nondeterministic*) *tree automaton* (DTA (*resp.*, NTA), for short) is a tuple $\langle \Sigma, \Lambda, \mathrm{Q}, \delta, q_I, \mathrm{Q_F} \rangle$, where all components but Λ and δ are defined as for a word automaton, $\Lambda \subseteq \mathbb{N}_+$ is the non-empty finite set of *node degrees*, and $\delta \colon \mathrm{Q} \times \Sigma \times \Lambda \to \mathrm{Q}^* \cup \{\bot, \top\}$ (*resp.*, $\delta \colon \mathrm{Q} \times \Sigma \times \Lambda \to 2^{\mathrm{Q}^*}$) is the *deterministic* (*resp.*, *nondeterministic*) *transition function* mapping each state $q \in \mathrm{Q}$, input symbol $\sigma \in \Sigma$, and node degree $d \in \Lambda$ to the tuple of successor states $\delta(q, \sigma, d) \in \mathrm{Q}^d$ (*resp.*, set of tuples of successor states $\delta(q, \sigma, d) \subseteq \mathrm{Q}^d$).

We only consider the Büchi acceptance condition, for both word and tree automata. The notions of *(accepting) run* and *recognised language* (denoted by $\mathrm{L}(\cdot)$) are the standard ones. For more details, we refer to [47,48].

Functions A *function signature* is a tuple $\mathcal{F} \triangleq \langle \mathrm{Fn}, \mathsf{ar} \rangle$, where Fn is a set of *function symbols* and $\mathsf{ar} \colon \mathrm{Fn} \to \mathbb{N}$ is an *arity function* mapping each symbol $f \in \mathrm{Fn}$ to its arity $\mathsf{ar}(f) \in \mathbb{N}$. An $\mathcal{F}$-structure $\mathfrak{F} \triangleq \langle \mathrm{D}, \cdot^{\mathfrak{F}} \rangle$ is defined by a *domain* D together with an *interpretation* of Fn over D, *i.e.*, every function symbol $f \in \mathrm{Fn}$ is interpreted in a function $f^{\mathfrak{F}} \colon \mathrm{D}^{\mathsf{ar}(f)} \to \mathrm{D}$. The set of terms built over the signature $\mathcal{F}$ and a set of variables Vr is denoted by Tr.

A *substitution* is a map $\mu \colon \mathrm{Vr} \to \mathrm{Tr}$ assigning a term to each variable; a *valuation w.r.t.* $\mathfrak{F}$ is a map $\xi \colon \mathrm{Vr} \to \mathrm{D}$ assigning an element of the domain to each variable. Given a term $t \in \mathrm{Tr}$, by t^μ we denote the *replacement* of all variables in t with the terms prescribed by the substitution μ; by $t^{\mathfrak{F},\xi}$ we denote the interpretation of t in $\mathfrak{F}$ under the valuation ξ, *i.e.*, the value assumed by t when each variable x is replaced with the value $\xi(x)$.

A set of terms $\mathrm{T} \subseteq \mathrm{Tr}$ *unifies* if there is a substitution μ such that $t_1^\mu = t_2^\mu$, for all $t_1, t_2 \in \mathrm{T}$. Similarly, T *equalises* in $\mathfrak{F}$ if there is a valuation ξ such that $t_1^{\mathfrak{F},\xi} = t_2^{\mathfrak{F},\xi}$, for all $t_1, t_2 \in \mathrm{T}$. For more details, we refer to [49,39].

3. Decidable fragments of strategy logic

Strategy Logic [25,27,50,36,28,29] extends LTL by allowing to *quantify* over strategies and to assign a strategy to each agent, by *binding* the latter with some quantified variable. We shall focus here on some relevant fragments of *One-Goal Strategy Logic* that restrict the syntax of the LTL component of the language.

In the following, a *quantifier prefix* will be a finite sequence $\wp \in \mathrm{Qn} \subseteq \{\mathrm{Q}x \mid x \in \mathrm{Vr}, \mathrm{Q} \in \{\exists, \forall\}\}^*$ of existential and universal quantifiers $\mathrm{Q}x$, in which variables $x \in \mathrm{Vr}$ occur at most once. With $\mathsf{vr}(\wp) \subseteq \mathrm{Vr}$ we denote the set of variables occurring in $\wp$. Similarly, a *binding prefix* is a finite sequence $\flat \in \mathrm{Bn} \subseteq (\mathrm{Ag} \times \mathrm{Vr})^{|\mathrm{Ag}|}$ of bindings of the form (a, x), in which each agent $a \in \mathrm{Ag}$ occurs exactly once.

3.1. One-goal strategy logic

One-Goal Strategy Logic is one of the largest decidable fragments of SL known to date and both satisfiability and model-checking problems are complete for 2ExpTime [28,29]. Its main constraint *w.r.t.* full SL is that bindings are tightly connected to quantifiers and agents cannot change strategies within the same sentence without quantifying on all of them again in a nested subsentence. For convenience, and *w.l.o.g.*, we provide below the positive normal form version of the language.

Definition 1 *(SL[1G] syntax).* SL[1G] *formulae* are generated from sets of literals $\mathrm{Lit} \triangleq \mathrm{AP} \cup \neg\mathrm{AP}$, quantifier prefixes Qn, and binding prefixes Bn via the following grammar, where $l \in \mathrm{Lit}$, $\wp \in \mathrm{Qn}$, and $\flat \in \mathrm{Bn}$, with $\mathsf{vr}(\wp) = \mathsf{vr}(\flat)$:

$$\varphi ::= \bot \mid \top \mid l \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \wp\flat\psi;$$
$$\psi ::= \varphi \mid \psi \wedge \psi \mid \psi \vee \psi \mid \mathtt{X}\,\psi \mid \psi\,\mathtt{U}\,\psi \mid \psi\,\mathtt{R}\,\psi.$$

SL[1G] denotes the set of sentences generated by Rule φ, while $\mathrm{FSL[1G]} \subset \mathrm{SL[1G]}$ identifies the associated *flat fragment*, *i.e.*, the subset generated by the variant of the grammar where l replaces φ within Rule ψ, *i.e.*, with ψ pure LTL.

SL[1G] sentences of the form $\wp\flat\psi$ are called *principal sentences*. An SL[1G] formula where each principal subsentence occurs at most once is called *simple*. Clearly, for any formula there exists an equivalent simple one that is obtained by standard variable renaming.

With $\mathsf{ap}(\varphi) \subseteq \mathrm{AP}$, $\mathsf{vr}(\varphi) \subseteq \mathrm{Vr}$, and $\mathsf{free}(\varphi) \subseteq \mathrm{Vr} \cup \mathrm{Ag}$ we denote, respectively, the sets of *atomic propositions*, *variables*, and *free variables and agents* occurring in φ. In addition, with $\mathsf{ap}^+(\varphi) \subseteq \mathsf{ap}(\varphi)$ we identify the subset of atomic propositions that only occur with positive polarity.

Being a first-order language, the semantics of SL formulae is defined *w.r.t.* an assignment, interpreting variables as strategies. This interpretation is extended to agents as well, in order to take care of bindings assigning strategies to agents. For a position v, let $\mathrm{Asg}(v) \triangleq (\mathrm{Vr} \cup \mathrm{Ag}) \rightharpoonup \mathrm{Str}(v)$ denote the set of such v-rooted assignments, *i.e.*, partial functions from variables and agents to strategies rooted at v. For a set $\mathrm{V} \subseteq (\mathrm{Vr} \cup \mathrm{Ag})$, we also provide, for convenience, the set of assignments defined precisely over V, *i.e.*, $\mathrm{Asg}(v, \mathrm{V}) \triangleq \{\chi \in \mathrm{Asg}(v) \mid \mathsf{dom}(\chi) = \mathrm{V}\}$, and those defined at least over V as $\mathrm{Asg}_{\subseteq}(v, \mathrm{V}) \triangleq \{\chi \in \mathrm{Asg}(v) \mid \mathrm{V} \subseteq \mathsf{dom}(\chi)\}$. As usual, given an assignment χ, a variable or agent $x \in (\mathrm{Vr} \cup \mathrm{Ag})$ and a strategy $\sigma \in \mathrm{Str}$, we denote with $\chi[x \mapsto \sigma]$ the assignment χ' resulting from assigning σ to x in χ.

In order to define the semantics of the temporal operators, we need to introduce a shift operator $\mathsf{sft}\colon (\mathrm{Ps} \times \mathrm{Asg}) \to (\mathrm{Ps} \times \mathrm{Asg})$ that, given the current position v and a v-rooted assignment χ computes the position v' reached after one steps along the play induced by χ starting from v and a v'-rooted assignment χ' assigning to each variable and agent the same strategy as χ but shifted one steps ahead along the play. Let $\pi = \mathsf{play}(\chi\restriction_{\mathrm{Ag}}, v)$ be the play from v induced by χ, where $\chi\restriction_{\mathrm{Ag}}$ corresponds to the strategy profile of χ. The shift function $\mathsf{sft}(v, \chi) \triangleq (v', \chi')$ is defined as follows:

- $v' = (\pi)_1$ is the position succeeding v in the play π;
- χ' is the assignment with the same domain as χ such that $\chi'(x)\colon \mathrm{Hst}(v') \to \mathrm{Ac}$ and $\chi'(x)(\rho) = \chi(x)(v \cdot \rho)$, for all $x \in \mathrm{Vr}$ and $\rho \in \mathrm{Hst}(v')$.

We define the i-step shift recursively as follows: $\mathsf{sft}^1(v, \chi) = \mathsf{sft}(v, \chi)$ and $\mathsf{sft}^{i+1}(v, \chi) = \mathsf{sft}(\mathsf{sft}^i(v, \chi))$.

Definition 2 *(SL semantics).* Given a CGS $\mathfrak{G} = \langle \mathrm{Ac}, \mathrm{Ps}, \tau, v_I, \lambda \rangle$, for all SL formulae φ, positions $v \in \mathrm{Ps}$, and v-rooted assignments $\chi \in \mathrm{Asg}_{\subseteq}(v, \mathsf{free}(\varphi))$, the satisfaction relation $\mathfrak{G}, (v, \chi) \models \varphi$ is inductively defined as follows:

1) $\mathfrak{G}, (v, \chi) \not\models \bot$ and $\mathfrak{G}, (v, \chi) \models \top$;
2) $\mathfrak{G}, (v, \chi) \models p$ (*resp.*, $\mathfrak{G}, (v, \chi) \models \neg p$), if $p \in \lambda(v)$ (*resp.*, $p \notin \lambda(v)$), for $p \in \mathrm{AP}$;
3) $\mathfrak{G}, (v, \chi) \models \phi_1 \vee \phi_2$, if $\mathfrak{G}, (v, \chi) \models \phi_1$ or $\mathfrak{G}, (v, \chi) \models \phi_2$;
4) $\mathfrak{G}, (v, \chi) \models \phi_1 \wedge \phi_2$, if $\mathfrak{G}, (v, \chi) \models \phi_1$ and $\mathfrak{G}, (v, \chi) \models \phi_2$;
5) For all $x \in \mathrm{Vr}$:
   a) $\mathfrak{G}, (v, \chi) \models \exists x.\, \phi$, if $\mathfrak{G}, (v, \chi[x \mapsto \sigma]) \models \phi$, for some strategy $\sigma \in \mathrm{Str}(v)$;
   b) $\mathfrak{G}, (v, \chi) \models \forall x.\, \phi$, if $\mathfrak{G}, (v, \chi[x \mapsto \sigma]) \models \phi$, for all strategies $\sigma \in \mathrm{Str}(v)$;
6) For all $a \in \mathrm{Ag}$ and $x \in \mathrm{Vr}$: $\mathfrak{G}, (v, \chi) \models (a, x)\phi$, if $\mathfrak{G}, (v, \chi[a \mapsto \chi(x)]) \models \phi$;
7) $\mathfrak{G}, (v, \chi) \models \mathtt{X}\,\phi$, if $\mathfrak{G}, \mathsf{sft}^1(v, \chi) \models \phi$;
8) $\mathfrak{G}, (v, \chi) \models \phi_1\,\mathtt{U}\,\phi_2$, if there exists an index $i \geq 0$ such that $\mathfrak{G}, \mathsf{sft}^i(v, \chi) \models \phi_2$ and $\mathfrak{G}, \mathsf{sft}^j(v, \chi) \models \phi_1$, for all indexes $0 \leq j < i$;
9) $\mathfrak{G}, (v, \chi) \models \phi_1\,\mathtt{R}\,\phi_2$ if, for all indexes $i \geq 0$, it holds that $\mathfrak{G}, \mathsf{sft}^i(v, \chi) \models \phi_2$ or $\mathfrak{G}, \mathsf{sft}^j(v, \chi) \models \phi_1$, for some $0 \leq j < i$.

For a sentence φ, we write $\mathfrak{G}, v \models \varphi$ instead of $\mathfrak{G}, (v, \varnothing) \models \varphi$ and, in addition, $\mathfrak{G} \models \varphi$ instead of $\mathfrak{G}, v_I \models \varphi$. As usual, we may use standard abbreviations such as, *e.g.*, $\psi_1 \to \psi_2$ for $\neg\psi_1 \vee \psi_2$ and $\mathtt{F}\,\psi$ for $\top\,\mathtt{U}\,\psi$.

The existence of a normal form for the models of sentences, as defined in the next section, relies on the notion of *skeleton* that breaks down their nesting structure. The idea is that a skeleton decomposes a sentence φ into a set Φ of simpler sentences of the flat fragment. Essentially, φ is stratified into layers whose sentences cannot occur within temporal operators. The connection between the layers is achieved by means of auxiliary atomic propositions, used as names of subsentences nested within temporal operators in the original formula. The skeleton is reminiscent of the classic decomposition technique used in the model-checking algorithms for CTL* [51,52].

Example 1. Consider the following sentence

$$\varphi \triangleq p \wedge \forall x \exists y \forall z (a,x)(b,y)(c,z)(\mathtt{X}\,(q \wedge (\mathtt{X}\,\mathtt{F}\,q)\,\mathtt{U}\,(\phi_1 \wedge \phi_2))), \text{ where}$$
$$\phi_1 \triangleq \exists x \forall y (a,x)(b,y)(c,y)(p\,\mathtt{U}\,q) \text{ and}$$
$$\phi_2 \triangleq \forall x \exists y (a,y)(b,x)(c,y)(\mathtt{G}\,\neg q).$$

The sentence φ can be stratified into two layers, using the fresh atomic propositions $\{s, s_1, s_2\}$ as names for the subsentences of φ. The mapping $\phi_1 \mapsto s_1$, $\phi_2 \mapsto s_2$ and $\forall x \exists y \forall z (a,x)(b,y)(c,z)(\mathtt{X}\,(q \wedge (\mathtt{X}\,\mathtt{F}\,q)\,\mathtt{U}\,(s_1 \wedge s_2))) \mapsto s$ provides one such stratification. In the end, the original formula φ is summarised by the positive Boolean formula $\zeta \triangleq p \wedge s$. □

The notion of skeleton is formalised by the following definition, where the relation $\prec$ encodes the ordering among the layers and the function ℓ assigns atomic propositions as names of subsentences of φ. We shall denote with $\mathsf{img}(\ell)$ the image of the function ℓ and with BF (*resp.*, BF$^+$) the set of Boolean (*resp.*, positive Boolean[1]) formulae over AP.

Definition 3 *(*SL[1G] *skeleton).* An SL[1G] *skeleton* is a tuple $\mathfrak{d} \triangleq \langle \zeta, \Phi, \ell \rangle$, where $\zeta \in \mathrm{BF}^+$ is a positive Boolean formula, $\Phi \subseteq \mathrm{FSL}[1\mathrm{G}]$ is a finite set of FSL[1G] principal sentences, and $\ell \colon \Phi \to \mathrm{AP}$ is an injective function mapping each sentence $\phi \in \Phi$ to an atomic proposition $\ell(\phi) \in \mathrm{AP}$ such that the following conditions hold true:

a) every atomic proposition $p \in \mathsf{img}(\ell)$ occurs in exactly one sentence $\phi \in \Phi \cup \{\zeta\}$ and at most once in it;
b) there exists a strict partial order $\prec \subseteq \Phi \times \Phi$ such that if $\ell(\phi) \in \mathsf{ap}(\phi')$ then $\ell(\phi) \in \mathsf{ap}^+(\phi')$ and $\phi \prec \phi'$, for all $\phi' \in \Phi$.

For a skeleton $\mathfrak{d}$, we denote with $\varphi_{\mathfrak{d}}$ the sentence derived from ζ by iteratively replacing each atomic proposition $p \in \mathsf{img}(\ell)$ with the corresponding FSL[1G] sentence $\ell^{-1}(p)$ until no atomic proposition in $\mathsf{img}(\ell)$ occurs in the sentence. This effectively reverts the stratification process described above. Note that the strict partial order $\prec$ on Φ ensures termination of the rewriting procedure. In addition, thanks to Item a), the resulting sentence is simple.

Proposition 1. *For each* SL[1G] *simple sentence φ, there exists a* SL[1G] *skeleton $\mathfrak{d}$ with $\varphi = \varphi_{\mathfrak{d}}$.*

Proof. Since φ is simple, any two principal subsentence of φ are different. Let Φ be the set of all such subsentences, each of which is simple as well. We first show that for any principal sentence ϕ in Φ there exists a pair (Φ_ϕ, ℓ_ϕ) with the properties of Definition 3. This is done by induction on the *principal sentence nesting depth* of the sentences of Φ, where $\mathsf{dep}(\phi)$, the principal sentences nesting depth of ϕ, is defined as follows:

$$\mathsf{dep}(\phi) = \begin{cases} 0, & \text{if } \phi \in \mathrm{LTL}; \\ 1 + \mathsf{dep}(\psi), & \text{if } \phi = \wp\flat\psi; \\ \mathsf{dep}(\psi), & \text{if } \phi = \mathtt{X}\,\psi; \\ \max\{\mathsf{dep}(\psi_1), \mathsf{dep}(\psi_2)\}, & \text{if } \phi = \psi_1 \circ \psi_2 \text{ with } \circ \in \{\wedge, \vee, \mathtt{U}, \mathtt{R}\}. \end{cases}$$

For the **base case**, we consider $\phi = \wp\flat\psi$ a smallest principal sentence with $\psi \in \mathrm{LTL}$. Therefore, $\mathsf{dep}(\phi) = 1$ and $\phi \in \mathrm{FSL}[1\mathrm{G}]$. In this case, we set $\Phi_\phi = \{\phi\}$ and $\ell_\phi(\phi) = p$, for some $p \in \mathrm{AP} \setminus \mathsf{ap}(\phi)$. The two properties of Definition 3 hold trivially for (Φ_ϕ, ℓ_ϕ).

For the **inductive case**, let $\phi = \wp\flat\psi$ with $\mathsf{dep}(\phi) > 1$. Let Δ contain the maximal principal subsentences of ψ, namely those which are not strict subsentence of other principal subsentences of ψ. By induction hypothesis, for each sentence $\phi' \in \Delta$ there exists a pair $(\Phi_{\phi'}, \ell_{\phi'})$ with the desired properties. We can also assume, *w.l.o.g.*, that the images of the naming functions $\ell_{\phi'}$ be all disjunct. Indeed, one can easily enforce this by renaming the atomic propositions in the image of each $\ell_{\phi'}$ and replacing their occurrences within $\Phi_{\phi'}$ with the fresh atomic propositions. This can clearly be done while still preserving all the properties of Definition 3.

Let $\widehat{\phi} \in \mathrm{FSL}[1\mathrm{G}]$ be the result of substituting in ϕ each principal subsentence $\phi' \in \Delta$ with its name $\ell_{\phi'}(\phi')$. At this point, we set $\Phi_\phi = \{\widehat{\phi}\} \cup \bigcup_{\phi' \in \Delta} \Phi_{\phi'}$ and $\ell_\phi = \{(\widehat{\phi}, p)\} \cup \bigcup_{\phi' \in \Delta} \ell_{\phi'}$, with p an atomic proposition not contained in the image of any

[1] Recall that a Boolean formula is positive if no negations occur.

$\ell_{\phi'}$. Since the sets $\Phi_{\phi'}$, for different ϕ', are all disjoint, the partial order of Item b can be taken as the union of the partial orders associated with each $(\Phi_{\phi'}, \ell_{\phi'})$, plus all the pairs $\{(\phi', \widehat{\phi}) \mid \phi' \in \Delta\}$, granting that $\widehat{\phi}$ is greater than each ϕ'. It is now a trivial exercise to show that the properties of Definition 3 are all satisfied for (Φ_ϕ, ℓ_ϕ) as well.

To complete the proof, let Δ be now the set of maximal principal sentences occurring in φ and, for each $\phi \in \Delta$, let (Φ_ϕ, ℓ_ϕ) be the pair satisfying the properties of Definition 3. Once again, we can assume that the images of the ℓ_ϕ are all disjoint. Let ζ be the Boolean formula obtained from φ by replacing each principal sentence in Δ with its name, $\Phi \triangleq \bigcup_{\phi\in\Delta} \Phi_\phi$, and $\ell \triangleq \bigcup_{\phi\in\Delta} \ell_\phi$. The resulting triple (ζ, Φ, ℓ) trivially satisfies all the required properties. □

Satisfaction of a skeleton $\eth$ by a CGS $\mathfrak{G}$ over the atomic propositions of $\eth$ is defined quite naturally. Specifically, the initial position of $\mathfrak{G}$ must locally satisfy the Boolean formula ζ, and any sentence in Φ, whose "name" labels a given position v of $\mathfrak{G}$, must be satisfied at v.

Definition 4 *(Skeleton satisfaction).* A CGS $\mathfrak{G} = \langle \mathrm{Ac}, \mathrm{Ps}, \tau, v_I, \lambda\rangle$ satisfies an SL[1G] skeleton $\eth = \langle \zeta, \Phi, \ell\rangle$, in symbols $\mathfrak{G} \models \eth$, if

1) $\lambda(v_I) \models \zeta$ and
2) $\mathfrak{G}, v \models \phi$, for all $\phi \in \Phi$ and $v \in \mathrm{Ps}$ with $\ell(\phi) \in \lambda(v)$.

The following result establishes the equisatisfiability of SL[1G] skeletons and their corresponding SL[1G] sentences. This generalises the corresponding result used in the model-checking procedure for CTL* [51]. In order to prove the result, we shall first provide, in the same vein as in [28], a Skolem-like semantics for principal sentences, which will greatly simplify the proof.

A *Skolem map* θ *for a quantifier prefix* $\wp$ is a function mapping any assignment for the universally quantified variables of $\wp$ to an assignment for all the variables in $\wp$ in such a way that:

- $\theta(\chi)(x) = \chi(x)$, for x a universal quantified variable in $\wp$;
- $\theta(\chi_1)(x) = \theta(\chi_2)(x)$, for any two assignments χ_1 and χ_2 for the universal variables of $\wp$ such that $\chi_1(y) = \chi_2(y)$, for each universal y preceding x in the quantifier prefix $\wp$.

Essentially, a Skolem map θ for $\wp$ establishes the response existential strategies to the universal ones in such a way that each existential strategy is chosen only depending on the universal strategies quantified before it in $\wp$. For a binding prefix $\wp\flat$ and a Skolem map θ for $\wp$, let $\theta_\flat$ be the extension of θ that, for every assignment χ and binding (a, x) in $\flat$, assigns to a the strategy $\theta(\chi)(x)$, *i.e.*, $\theta_\flat(\chi)(a) = \theta(\chi)(x)$. It was proven in [28, Theorem 4.5] that $\mathfrak{G}, v \models \wp\flat\psi$ *iff* $\mathfrak{G}, (v, \theta_\flat(\chi)) \models \psi$, for some Skolem map θ for $\wp$ and every assignment $\chi \in \mathrm{Asg}(v, \mathrm{V})$, with V the set of universal variables of $\wp$.

Theorem 1. $\varphi_\eth$ *is satisfiable* iff $\eth$ *is satisfied by a tree* CGS*, for every* SL[1G] *skeleton* $\eth$*.*

Proof. For the **if direction**, we first prove that $\mathfrak{G}, v \models \phi_\eth$, for all $\phi \in \Phi$ and $v \in \mathrm{Ps}$ with $\ell(\phi) \in \lambda(v)$, by induction on the depth of ϕ with respect to the order of $\eth$. For the **base case**, assume the depth of ϕ *w.r.t.* $\prec$ be 0, namely ϕ is a pure LTL formula. Then, $\phi = \phi_\eth$ and the thesis follows directly by Condition 2 of Definition 4. For the **inductive case**, let the depth of $\phi = \wp\flat\psi$ *w.r.t.* $\prec$ be k. For every LTL (flat) formula η, within which only names of principal sentences of order strictly less than k occur, and every assignment $\chi \in \mathrm{Asg}(v, \mathrm{Ag})$, it holds that if $\mathfrak{G}, (v, \chi) \models \eta$ then $\mathfrak{G}, (v, \chi) \models \eta_\eth$. This implication can be proven by a standard structural induction, recalling that names of principal sentences can only appear with positive polarity in η. Consider now a position v with $\ell(\phi) \in \lambda(v)$. By Condition 2 of Definition 4, we know that $\mathfrak{G}, v \models \phi$. Hence, there exists a Skolem map θ for $\wp$ such that $\mathfrak{G}, (v, \theta_\flat(\chi)) \models \psi$, for all assignments $\chi \in \mathrm{Asg}(v, \mathrm{V})$, with V the set of universally quantified variables of $\wp$. Observe that ψ is a pure LTL formula where only names of principal subsentences of order less than k can occur. Hence, by the property shown above for LTL formulae, we can conclude $\mathfrak{G}, (v, \theta_\flat(\chi)) \models \psi_\eth$, which gives us $\mathfrak{G}, v \models \wp\flat\psi_\eth$ as desired.

To conclude the proof of the **if direction**, take the Boolean formula ζ of the skeleton $\eth$. By assumption, it holds $\mathfrak{G}, v_I \models \zeta$. Let $\Delta = \{\phi \in \Phi \mid \ell(\phi) \in \lambda(v_I)\}$ be the set of principal sentences of Φ occurring in ζ. By the result just proven, we have that $\mathfrak{G}, v_I \models \phi_\eth$, for each $\phi \in \Delta$. By definition, $\varphi_\eth$ is obtained from ζ by replacing each name $\ell(\phi)$, for $\phi \in \Delta$, with $\phi_\eth$. The conclusion $\mathfrak{G}, v_I \models \varphi_\eth$, then, immediately follows.

In order to prove the **only-if direction**, we first prove the following auxiliary property. Let $\mathfrak{G} = \langle \mathrm{Ac}, \mathrm{Ps}, \tau, v_I, \lambda\rangle$ be a CGS satisfying $\mathfrak{G}, v \models \phi$ *iff* $\mathfrak{G}, v \models \ell(\phi)$, for all $v \in \mathrm{Ps}$ and $\phi \in \Phi$. Then $\mathfrak{G}, v \models \phi$ *iff* $\mathfrak{G}, v \models \phi_\eth$, for all $v \in \mathrm{Ps}$ and $\phi \in \Phi$. We proceed by induction the depth of the strict partial order $\prec$ underlying $\eth$. The **base case** where ϕ has depth 0, hence does not contain names of other principal sentences in Φ, is trivial, since we have $\phi = \phi_\eth$ in this case. For the **inductive case**, let ϕ have depth $k > 1$ and $\Delta = \{\phi' \mid \ell(\phi') \text{ occurs in } \phi\}$. Clearly, each element $\phi' \in \Delta$ has depth strictly smaller than ϕ and, as a consequence, by inductive hypothesis we conclude that $\mathfrak{G}, v \models \phi'$ *iff* $\mathfrak{G}, v \models \phi'_\eth$, for all $v \in \mathrm{Ps}$. In addition, by the assumption on $\mathfrak{G}$, we know that $\mathfrak{G}, v \models \phi'$ *iff* $\mathfrak{G}, v \models \ell(\phi')$, for all $v \in \mathrm{Ps}$. It, then, follows that $\mathfrak{G}, v \models \phi'_\eth$ *iff* $\mathfrak{G}, v \models \ell(\phi')$, for all $v \in \mathrm{Ps}$. Now, observe that $\phi_\eth = \phi\,[\delta]$, where $\delta \triangleq \{\ell(\phi') \mapsto \phi'_\eth \mid \phi' \in \Delta\}$ is the substitution replacing each name $\ell(\phi')$ of

a sentence ϕ' in Δ with the equivalent (relative to $\mathfrak{G}$) fully expanded version $\phi'_{\mathfrak{d}}$. By substitution of equivalents we obtain that $\mathfrak{G}, v \models \phi$ *iff* $\mathfrak{G}, v \models \phi_{\mathfrak{d}}$, for all positions $v \in \mathrm{Ps}$.

To conclude the proof of the **only-if direction**, assume now that $\mathfrak{G} = \langle \mathrm{Ac}, \mathrm{Ps}, \tau, v_I, \lambda \rangle$ is a satisfying CGS for $\varphi_{\mathfrak{d}}$. Let $\mathfrak{G}' = \langle \mathrm{Ac}, \mathrm{Ps}, \tau, v_I, \lambda' \rangle$ be the CGS obtained from $\mathfrak{G}$ by setting $\lambda'(v) = \lambda(v) \cup \{\ell(\phi) \mid \phi \in \Phi \text{ and } \mathfrak{G}, v \models \phi\}$. Clearly, $\mathfrak{G}'$ still satisfies $\varphi_{\mathfrak{d}}$ and, by construction, also satisfies $\mathfrak{G}, v \models \phi$ *iff* $\mathfrak{G}, v \models \ell(\phi)$, for all $v \in \mathrm{Ps}$ and $\phi \in \Phi$. Observe that this last property implies Condition 2 of Definition 4. Now, by applying the above result, we obtain that $\mathfrak{G}', v \models \phi$ *iff* $\mathfrak{G}', v \models \phi_{\mathfrak{d}}$, for all $\phi \in \Phi$. Let $\Delta = \{\phi' \mid \ell(\phi') \text{ occurs in } \zeta\}$ and $\delta = \{\phi' \mapsto \phi'_{\mathfrak{d}} \mid \phi' \in \Delta\}$ be a substitution mapping every sentence in Δ to an equivalent (relative to $\mathfrak{G}'$) maximal strict subsentence of $\varphi_{\mathfrak{d}}$. Since $\varphi_{\mathfrak{d}} = \zeta[\delta]$, we immediately obtain that $\mathfrak{G}', v_I \models \zeta$, hence also Condition 1 of Definition 4 holds and we get the thesis. □

3.2. Non-recurrent one-goal strategy logics

The main source of complexity for SL[1G], or CTL* and ATL* for that matter, resides in the ability to express properties that request satisfaction of a given sentence an unbounded number of times along a computation, as, *e.g.*, in the CTL formula $\mathtt{EG}(\neg p \wedge \mathtt{EX}\, p)$ that states the existence of a path all of whose states falsify p but must have an adjacent state satisfying p that does not belong to that path. Clearly, any model that satisfies the formula must satisfy $\mathtt{EX}\, p$ an infinite number of times along the witness path. Given the branching nature of quantifications in SL, this may lead, very much like in the case of CTL*, to models with an unbounded number of branching points. In general, such models can be recognised by tree automata with a doubly exponential number of states [53,36]. Emptiness for tree automata is known to be PTime-hard [54], which leads to the 2ExpTime-hardness for deciding SL[1G].

To avoid this issue, we restrict the number of times a given sentence can be requested, by preventing sentences in the left-hand (*resp.*, right-hand) argument of the until (*resp.*, release) operator. We call the resulting fragment *non-recurrent*, in that it forbids an unbounded number of requests of the same sentence along a computation.

Definition 5 *(*SL[1G] *fragments).* Formulas of *non-recurrent fragments* of SL[1G] are generated from the sets of literals Lit, quantifier prefixes Qn, and binding prefixes Bn via the following grammar, with $l \in \mathrm{Lit}$, $\psi \in \mathrm{LTL(AP)}$, $\beta \in \mathrm{BF(AP)}$, $\wp \in \mathrm{Qn}$, and $\flat \in \mathrm{Bn}$ such that $\mathsf{vr}(\wp) = \mathsf{vr}(\flat)$:

$$
\begin{aligned}
\mathrm{SL}^{/\!\!0}[1\mathrm{G}]\colon\quad & \varphi \coloneqq \bot \mid \top \mid l \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \wp^{\flat}\eta \mid \wp^{\flat}\psi;\\
& \eta \coloneqq \varphi \mid \eta \wedge \eta \mid \eta \vee \eta \mid \psi\, \mathtt{U}\, \varphi \mid \varphi\, \mathtt{R}\, \psi \mid \mathtt{X}\eta \mid \mathtt{X}\psi;\\
\mathrm{WSL}^{/\!\!0}[1\mathrm{G}]\colon\quad & \varphi \coloneqq \bot \mid \top \mid l \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \wp^{\flat}\eta;\\
& \eta \coloneqq \varphi \mid \eta \wedge \eta \mid \eta \vee \eta \mid \beta\, \mathtt{U}\, \varphi \mid \varphi\, \mathtt{R}\, \beta \mid \mathtt{X}\eta.
\end{aligned}
$$

$\mathrm{FSL}^{/\!\!0}[1\mathrm{G}] \subset \mathrm{SL}^{/\!\!0}[1\mathrm{G}]$ and $\mathrm{FWSL}^{/\!\!0}[1\mathrm{G}] \subset \mathrm{WSL}^{/\!\!0}[1\mathrm{G}]$ identify the corresponding *flat fragments*, *i.e.*, the subsets generated by the variants of the grammars where l replaces φ within Rule η.

For each fragment, the rule φ takes care of the first-order (branching) structure of the language, while the rule η handles the temporal portion. The non-recurrence constraint is embedded in the cases for the until and release operators within the rule η, restricting the left-hand (*resp.*, right-hand) argument of the until (*resp.*, release) operators to be a pure LTL formula with no nesting of sentences. The weak fragment further restricts those arguments so that no temporal operators can occur altogether, *i.e.*, they can only accommodate Boolean formulae. No restriction is imposed on the next operator, while negation can only be applied to atomic propositions in AP.

In the rest of this work, we call *Weak* LTL (WLTL for short), the following extension of the fragment of LTL that agrees with Rule η of $\mathrm{FWSL}^{/\!\!0}[1\mathrm{G}]$:

$$
\begin{aligned}
& \eta \coloneqq \bot \mid \top \mid l \mid \eta \wedge \eta \mid \eta \vee \eta \mid \upsilon\, \mathtt{U}\, \upsilon \mid \upsilon\, \mathtt{R}\, \upsilon \mid \mathtt{X}\eta;\\
& \upsilon \coloneqq \bot \mid \top \mid l \mid \upsilon \wedge \upsilon \mid \upsilon \vee \upsilon \mid \mathtt{X}\upsilon.
\end{aligned}
$$

Restrictions similar to those of WLTL have been provided in [40] and shown to be useful in the definition of temporal logics over data [41,42], analysis of pushdown systems [43], and in synthesis of hybrid systems [44].

The following example illustrates a possible application of the non-recurrent fragment in the context of exchange protocol design.

Example 2. Consider a *fairness exchange protocol*, where two agents seek to exchange a piece of information m^* in a fair fashion, *i.e.*, in such a way that either both agents have the entire requested information or none of them has it at the end of the exchange. In the presence of unreliable or insecure channels, this can be achieved by means of a trusted third party (TTP) that can be interrogated by one of the agents in case of unfair behaviour by the other party. Such a protocol can be modelled as a game played by the two agents, an implicit reactive agent TTP and an intruder agent modelling the unreliable channel. We require that in the resulting multi-agent system the TTP can ensure the fairness property, regardless of how the insecure channel among the agents behaves.

We want to synthesise an actual protocol that achieves the fairness requirement and comprises a bounded number of steps, let us say k, which is a standard assumption in communication protocols. The situation can be modelled as the conjunction of the following $\mathrm{WSL}^{\not\emptyset}[1\mathrm{G}]$ formulae, where $\mathtt{A}$, $\mathtt{B}$, $\mathtt{I}$, and $\mathtt{T}$ correspond to the two parties, the intruder, controlling the communication channel between them, and the TTP, respectively.

Let the atomic proposition $\mathtt{k}^m_\alpha$ (*resp.*, $\mathtt{r}^m_\alpha$) stand for "α *knows (*resp.*, receives) message* m" in the current step, and proposition $\mathtt{s}^m_{\alpha\beta}$ stand for "α *sends message* m *to* β". Moreover, let $\flat \triangleq (\mathtt{A}, x_\mathtt{A})(\mathtt{B}, x_\mathtt{B})(\mathtt{I}, y)$, $\overline{\mathtt{A}} = \mathtt{B}$ and $\overline{\mathtt{B}} = \mathtt{A}$. Then the following formulae encode the constraints on the agents' behaviour, as well as the fairness requirement, in a single step of the protocol. Here, we abstract from possible encryption mechanisms. We also assume monotonicity of agents' knowledge and of the sent/received messages, which can easily be encoded in LTL.

1. $\phi^{\alpha,m}_{agn} \triangleq \bigwedge_{\beta\in\{\overline{\alpha},\mathtt{T}\}}\left(\mathtt{k}^m_\alpha \rightarrow \exists x_\alpha \forall x_{\overline{\alpha}} \forall y \flat \mathtt{X}\, \mathtt{s}^m_{\alpha\beta}\right)$: if agent α currently knows message m, it can send it to $\beta \in \{\overline{\alpha}, \mathtt{T}\}$, regardless of what the other agents do;
2. $\phi^{\alpha,m}_{int+} \triangleq \left(\mathtt{k}^m_\mathtt{I} \rightarrow \exists y \forall x_\mathtt{A} \forall x_\mathtt{B} \flat \mathtt{X}\, \mathtt{r}^m_\alpha\right)$: if the intruder knows message m, it has a way to forward it to agent α, regardless of the other agents;
3. $\phi^{\alpha,m}_{int-} \triangleq \left(\neg \mathtt{s}^m_{\mathtt{T}\alpha} \rightarrow \exists y \forall x_\mathtt{A} \forall x_\mathtt{B} \flat \mathtt{X} \neg \mathtt{r}^m_\alpha\right)$: the intruder has a way to prevent agent α from receiving the message m, unless the TTP intervenes to send it to α;
4. $\phi^{\alpha,m^*}_{frn} \triangleq (\mathtt{r}^{m^*}_\alpha \rightarrow \exists x_{\overline{\alpha}} \forall x_\alpha \forall y \flat \mathtt{F}\, \mathtt{r}^{m^*}_{\overline{\alpha}})$: if agent α has already received the message m^*, then agent $\overline{\alpha}$ has a way to eventually obtain the message, regardless of the other agents.

Let us set $\mathtt{G}^0(\varphi) \triangleq \varphi$ and $\mathtt{G}^{k+1}(\varphi) \triangleq \varphi \wedge \mathtt{X}\,\mathtt{G}^k(\varphi)$. The formula $\mathtt{G}^k(\varphi)$ requires φ to hold in the next k steps of a computation at least. Moreover, let

$$\mathtt{AG}^k(\varphi) \triangleq \forall x_\mathtt{A} \forall x_\mathtt{B} \forall y \flat \mathtt{G}^k(\varphi)$$

be the formula requiring $\mathtt{G}^k(\varphi)$ along all the paths induced by the possible choices of the agents starting from the current state onward. Then, by iterating the above constraints, the following $\mathrm{WSL}^{\not\emptyset}[1\mathrm{G}]$ formula encodes a k-step protocol with fairness constraint on m^*:

$$\varphi^k_{prt} \triangleq \mathtt{AG}^k\left(\bigwedge\nolimits_{\alpha\in\{\mathtt{A},\mathtt{B}\}}\left(\bigwedge\nolimits_{m\in\mathrm{Msg}}(\phi^{\alpha,m}_{agn} \wedge \phi^{\alpha,m}_{int+} \wedge \phi^{\alpha,m}_{int-}) \wedge \phi^{\alpha,m^*}_{frn}\right)\right). \quad \square$$

Obviously, we can obtain skeletons for the new fragments, by suitably restricting their components to the corresponding flat fragments. An $\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]$ (*resp.*, $\mathrm{WSL}^{\not\emptyset}[1\mathrm{G}]$) skeleton $\eth = \langle \zeta, \Phi, \ell\rangle$ is a principal SL[1G] skeleton such that $\Phi \subseteq \mathrm{FSL}^{\not\emptyset}[1\mathrm{G}]$ (*resp.*, $\Phi \subseteq \mathrm{FWSL}^{\not\emptyset}[1\mathrm{G}]$). The analogous of Proposition 1 holds for the two new fragments $\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]$ and $\mathrm{WSL}^{\not\emptyset}[1\mathrm{G}]$ as well.

Proposition 2. *Each* $\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]$ *(*resp.*,* $\mathrm{WSL}^{\not\emptyset}[1\mathrm{G}]$*) sentence* φ *enjoys an* $\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]$ *(*resp.*,* $\mathrm{WSL}^{\not\emptyset}[1\mathrm{G}]$*) skeleton* $\eth$ *with* $\varphi = \varphi_\eth$.

The constraint on the non-recurrence of sentences allows us to strengthen Theorem 1 and show that a sentence is satisfiable *iff* its skeleton can be satisfied by a model where each subsentence is requested at most once. This property is formalised by the definition of *single-time satisfaction* and the following theorem. The result is instrumental to the definition of normal models (see next section) and, ultimately, to the main complexity results.

Definition 6 *(Single-time skeleton satisfaction).* A CGS $\mathfrak{G} = \langle \mathrm{Ac}, \mathrm{Ps}, \tau, v_I, \lambda\rangle$ *single-time satisfies* a skeleton $\eth = \langle \zeta, \Phi, \ell\rangle$ if

1. $\mathfrak{G} \models \eth$ and
2. $\ell(\phi) \in \lambda(v)$ implies $\ell(\phi) \notin \lambda(w)$, for all $\phi \in \Phi$, $v \in \mathrm{Ps}$, and $w \in \tau^+(v)$.

Theorem 2. $\varphi_\eth$ *is satisfiable* iff $\eth$ *is single-time satisfied by a tree* CGS*, for every* $\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]$ *skeleton* $\eth$.

Proof. Assume $\varphi_\eth$ is satisfiable. By Theorem 1, $\eth$ is satisfiable as well. Thus, let $\mathfrak{G}$ be one of the tree CGSs satisfying $\eth$. The proof proceeds by induction on the depth k of the strict partial order $\prec$ underlying the skeleton $\eth$. The idea is the following: starting from $\mathfrak{G}$, we perform a sequence $\mathfrak{G}_{k+1}$, $\mathfrak{G}_k$, $\mathfrak{G}_{k-1}$, …, $\mathfrak{G}_0$ of structure-preserving model transformations, where $\mathfrak{G}_{k+1} \triangleq \mathfrak{G}$. The labelling of the models is modified in such a way that all sentences in Φ at level i in the ordering $\prec$ are single-time satisfied in $\mathfrak{G}_j$, for every $j \leq i$. More specifically, sentences at level k only need to be verified at the root, while those at level $i < k$ need only be checked at the first occurrence of a witness of the until/release operator containing it. Hence, the labelling of $\mathfrak{G}_i$ is obtained from $\mathfrak{G}_{i+1}$, by removing, along each path, every occurrence of the name of a sentence at level i except for the one that serves as witness of the corresponding until/release operator. By construction, the name $\ell(\phi)$ of every sentence ϕ in Φ occurs only once along any path of $\mathfrak{G}_0$. Hence, $\mathfrak{G}_0$ single-time satisfies $\eth$. $\square$

4. Normal models for $\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]$

The efficient solution of the satisfiability problem for the non-recurrent fragments relies on the fact that any of their sentences is satisfiable by models of a specific structure, namely, by bounded-fork tree CGS. This can be proven by first extending SL with function symbols, which allows for bindings containing strategy terms. This will enable us to state a Skolem normal-form theorem for SL[1G]. We will leverage this result to show that any model for a sentence φ of $\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]$ in Skolem form can be transformed into a bounded-fork tree satisfying φ, where forks only occur as a result of non-unifying strategy terms within the bindings of φ.

4.1. Functions in SL

Given a function signature $\mathcal{F}$, with $\mathrm{SL}[1\mathrm{G},\mathcal{F}]$ we denote the extension of SL[1G] where we allow to bind agents with complex terms instead of just simple variables. This means that the set of bindings Bn in the syntax gets replaced by its extension $\mathrm{Bn}(\mathcal{F}) \subseteq (\mathrm{Ag} \times \mathrm{Tr})^{|\mathrm{Ag}|}$. A binding prefix is, thus, a finite sequence $\flat \in \mathrm{Bn}(\mathcal{F})$ of bindings (a,t), with $t \in \mathrm{Tr}$, in which each agent $a \in \mathrm{Ag}$ occurs exactly once. $\forall\mathrm{SL}[1\mathrm{G},\mathcal{F}]$ represents the universal fragment of $\mathrm{SL}[1\mathrm{G},\mathcal{F}]$, where existential quantifiers are forbidden. For convenience, in the following we may use the notation $\forall\flat\varphi$ as a shorthand for the $\forall\mathrm{SL}[1\mathrm{G},\mathcal{F}]$ sentence $\forall x_1 \forall x_2 \ldots \forall x_k \flat \varphi$, where $\{x_1, x_2, \ldots, x_k\}$ is the set of variables occurring in the binding prefix $\flat$.

In order to define the semantics of an $\mathrm{SL}[1\mathrm{G},\mathcal{F}]$ sentence, we need to provide a *strategy interpretation* for all function symbols in Fn. This can be done by means of a map $\Im\colon v \in \mathrm{Ps} \mapsto \mathfrak{F}_v$ that assigns to each position v an $\mathcal{F}$-structure $\mathfrak{F}_v = \langle \mathrm{Str}(v), \cdot^{\mathfrak{F}_v} \rangle$, whose domain $\mathrm{Str}(v)$ is the set of strategies rooted at v. Given a pair $(\mathfrak{G},\Im)$ of a CGS $\mathfrak{G}$ and a strategy interpretation $\Im$, called *interpreted* CGS, we can define the satisfaction relation $(\mathfrak{G},\Im),(v,\chi) \models \varphi$ as in Definition 2, where, however, Item 6 is replaced by the following clause.

- For all $a \in \mathrm{Ag}$ and $t \in \mathrm{Tr}$: $(\mathfrak{G},\Im),(v,\chi) \models (a,t)\phi$, if $(\mathfrak{G},\Im),(v,\chi') \models \phi$, where $\chi' \triangleq \chi[a \mapsto t^{\Im(v),\chi}]$.

Observe that in the new condition agent $a \in \mathrm{Ag}$ is bound to the strategy $t^{\Im(v),\chi}$, namely, the interpretation of the term t under the v-rooted assignment $\chi \in \mathrm{Asg}(v)$ in the $\mathcal{F}$-structure $\Im(v) = \langle \mathrm{Str}(v), \cdot^{\Im(v)} \rangle$ associated with v. Intuitively, we assign to each agent a a strategy dependent on the strategies associated with the variables occurring in the term t.

An $\mathrm{SL}[1\mathrm{G},\mathcal{F}]$ sentence φ is *satisfied* by a CGS $\mathfrak{G}$, in symbols $\mathfrak{G} \models \varphi$, if there exists a strategy interpretation $\Im$ such that $(\mathfrak{G},\Im) \models \varphi$, where the latter stands for $(\mathfrak{G},\Im),(v_I,\varnothing) \models \varphi$. In the rest of the work, with $\mathsf{skm}\colon \mathrm{SL}[1\mathrm{G}] \to \forall\mathrm{SL}[1\mathrm{G},\mathcal{F}]$ we denote the function mapping each SL[1G] sentence φ to the corresponding *Skolem normal-form* $\mathsf{skm}(\varphi)$, where each variable x existentially quantified in a subsentence ϕ of φ is replaced by a fresh function symbol applied to the variables universally quantified in ϕ before x.

Example 3. Let φ be the SL[1G] sentence used in Example 1 to exemplify the notion of SL[1G] skeleton. We have that

$$\mathsf{skm}(\varphi) = p \wedge \forall x \forall z (a,x)(b,f_1(x))(c,z)(\mathtt{X}\,(q \wedge (\mathtt{X}\,\mathtt{F}\,q)\,\mathtt{U}\,(\mathsf{skm}(\phi_1) \wedge \mathsf{skm}(\phi_2)))),$$

where

$$\begin{aligned} \mathsf{skm}(\phi_1) &= \forall y (a,f_2)(b,y)(c,y)(p\,\mathtt{U}\,q) \text{ and} \\ \mathsf{skm}(\phi_2) &= \forall x (a,f_3(x))(b,x)(c,f_3(x))(\mathtt{G}\neg q). \end{aligned}$$

In φ, the existential variable y of the outermost sentence is replaced by the term $f_1(x)$, since the strategy chosen by agent b only depends on the strategy used by agent a. A similar reasoning applies to the subsentence ϕ_2. In ϕ_1, instead, the existential variable x is replaced by the constant f_2, as the strategy for agent a does not depend on those of b and c. □

In [28] (see Theorem 4.5 and Corollary 4.6), it has been proved that SL enjoys a semantic version of the *Skolem normal-form theorem*, where the interpretation of Skolem functions given by a Skolem map (called *Skolem dependence function* in [28]) is given at the meta-level. Thanks to the introduction of function symbols in the syntax of the logic, this result can now be stated at the object-level in the classic way. Indeed, from the Skolem maps of the quantifier prefixes in an SL[1G] sentence φ, one can extract the strategy interpretation of all the function symbols in the $\forall\mathrm{SL}[1\mathrm{G},\mathcal{F}]$ sentence $\mathsf{skm}(\varphi)$ and, *vice versa*, from the strategy interpretation satisfying $\mathsf{skm}(\varphi)$, one can reconstruct the Skolem maps for φ.

Theorem 3. *Let $\mathfrak{G}$ be a* CGS. *An* SL[1G] *sentence φ is satisfied by $\mathfrak{G}$* iff *the* $\forall\mathrm{SL}[1\mathrm{G},\mathcal{F}]$ *sentence $\mathsf{skm}(\varphi)$ is satisfied by $\mathfrak{G}$.*

A fundamental property of SL[1G], which allows both its model-checking and satisfiability problems to be elementarily decidable, is that every satisfiable sentence of this logic is *behaviourally satisfiable* (see Theorem 4.20 and Corollary 4.21 in [28]). Essentially, this means that each action chosen by an agent, for some history of a play, only depends on the actions chosen by the other agents along that history. In other words, an agent does not need to forecast the future to play optimally. At this point, we can formalise this intuition and restate the result proven in [28] as follows.

We say that two strategies $\sigma_1, \sigma_2 \in \mathrm{Str}(v)$ are *equal at* a history $\rho \in \mathrm{Hst}(v)$ (ρ-*equal*, for short), if $\sigma_1(\rho) = \sigma_2(\rho)$. This notion immediately lifts to vectors of strategies $\vec{\sigma}_1, \vec{\sigma}_2 \in \mathrm{Str}(v)^k$, for some $k \in \mathbb{N}$, as usual: $\vec{\sigma}_1$ and $\vec{\sigma}_2$ are ρ-*equal* if all their k components $(\vec{\sigma}_1)_i$ and $(\vec{\sigma}_2)_i$ are ρ-equal, with $i \in [0, k)$.

A function over strategies $f : \mathrm{Str}(v)^k \to \mathrm{Str}(v)$, for some arity $k \in \mathbb{N}$, is *behavioural* if, for every history $\rho \in \mathrm{Hst}$ and pair of ρ-equal k-vectors of strategies $\vec{\sigma}_1, \vec{\sigma}_2 \in \mathrm{Str}(v)^k$, it holds that $f(\vec{\sigma}_1)(\rho) = f(\vec{\sigma}_2)(\rho)$. Basically, when given as input two ρ-equal tuples of strategies, a behavioural function f produces a strategy that always chooses the same value on the history ρ. A strategy interpretation $\Im$ *w.r.t.* a given CGS $\mathfrak{G}$ is *behavioural* if the function $f^{\Im(v)}$ is behavioural, for every position $v \in \mathrm{Ps}$ and symbol $f \in \mathrm{Fn}$. An $\mathrm{SL}[1\mathrm{G}, \mathcal{F}]$ sentence φ is *behaviourally satisfied* by a CGS $\mathfrak{G}$ if there exists a behavioural strategy interpretation $\Im$ such that $(\mathfrak{G}, \Im) \models \varphi$. The following result can be proven by exploiting Corollary 4.21 in [28] and observing that the interpretation function $\Im$ semantically corresponds to the behavioural Skolem map used in the statement of the corollary.

Theorem 4. *For any* CGS $\mathfrak{G}$ *and* $\mathrm{SL}[1\mathrm{G}]$ *sentence* φ*, the sentence* $\mathsf{skm}(\varphi)$ *is satisfied by* $\mathfrak{G}$ iff *it is behaviourally satisfied by* $\mathfrak{G}$.

Proof. Corollary 4.21 of [28] (see also Definitions 4.7, 4.10, and Lemma 4.8) states that an $\mathrm{SL}[1\mathrm{G}]$ sentence $\wp\flat\psi$ is satisfied at position v of a CGS $\mathfrak{G}$, *i.e.*, $\mathfrak{G}, v \models \wp\flat\psi$, *iff* it is behaviourally satisfied at v in $\mathfrak{G}$, in the sense that there exists a behavioural Skolem map θ for $\wp$, such that $\mathfrak{G}, (v, \theta(\chi)) \models \flat\psi$, for all assignments $\chi \in \mathrm{Asg}(v, \mathrm{V})$, with V the set of universal variables of $\wp$. We recall that a Skolem map is behavioural if $\theta(\chi_1)(x)(\rho) = \theta(\chi_2)(x)(\rho)$, for any history $\rho \in \mathrm{Hst}(v)$ and two assignments χ_1 and χ_2 for the universal variables of $\wp$ such that $\chi_1(y)(\rho) = \chi_2(y)(\rho)$, for each universal y preceding x in the quantifier prefix $\wp$. The notion of behaviouralness just described practically coincide with the one introduced above for strategy interpretations. Hence, it is immediate to see that, for every behavioural Skolem map θ, there exists a behavioural strategy interpretation $\Im$ and, *vice versa*, for every behavioural strategy interpretation $\Im$, there exists a behavioural Skolem map θ, such that $\mathfrak{G}, (v, \theta(\chi)) \models \flat\psi$ *iff* $(\mathfrak{G}, \Im), (v, \chi') \models \flat'\psi$, for all $\chi \in \mathrm{Asg}(v, \mathrm{V})$, where $\flat'$ is the binding of $\mathsf{skm}(\wp\flat\psi)$. Intuitively, for every existential variable x in $\wp\flat\psi$ and corresponding function f in $\mathsf{skm}(\wp\flat\psi)$, one can use θ evaluated at x to define the interpretation of f in $\Im$ and, *vice versa*, one can use the interpretation of f in $\Im$ to define the value of θ for x. Now, by Theorem 3, $\mathfrak{G}, v \models \wp\flat\psi$ *iff* $(\mathfrak{G}, \Im), v \models \mathsf{skm}(\wp\flat\psi)$, for some strategy interpretation $\Im$. At this point, by combining this with the result of [28] and the observations above, we obtain that $(\mathfrak{G}, \Im), v \models \mathsf{skm}(\wp\flat\psi)$, for some strategy interpretation $\Im$, *iff* $(\mathfrak{G}, \Im'), v \models \mathsf{skm}(\wp\flat\psi)$, for some behavioural strategy interpretation $\Im'$, which concludes the proof. □

4.2. Unifying bindings & paths

In [37] it has been observed that the decidability of the satisfiability problem for $\mathrm{SL}[1\mathrm{G}]$ can be attributed to the fact that bindings in that fragment indivisibly associate variables with agents. Here we further exploit that observation to define a normal form for $\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]$ models, by applying the notions of *Herbrand property* and *quasi-Herbrand structures* devised in [39], so that unifying bindings identify the same paths.

The notion of $\mathrm{SL}[1\mathrm{G}]$ (*resp.*, $\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]$) skeleton, as well as the corresponding concept of (*resp.*, single-time) skeleton satisfaction, immediately lifts to $\mathrm{SL}[1\mathrm{G}, \mathcal{F}]$ (*resp.*, $\mathrm{SL}^{\not\emptyset}[1\mathrm{G}, \mathcal{F}]$) in the obvious way. A skeleton is universal if all formulas in Φ are universal, *i.e.*, $\Phi \subseteq \forall\mathrm{SL}[1\mathrm{G}, \mathcal{F}]$. Given an $\mathrm{SL}[1\mathrm{G}]$ (*resp.*, $\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]$, $\mathrm{WSL}^{\not\emptyset}[1\mathrm{G}]$) skeleton $\eth$, we denote with $\mathsf{skm}(\eth)$ the (universal) $\forall\mathrm{SL}[1\mathrm{G}, \mathcal{F}]$ (*resp.*, $\forall\mathrm{SL}^{\not\emptyset}[1\mathrm{G}, \mathcal{F}]$, $\forall\mathrm{WSL}^{\not\emptyset}[1\mathrm{G}, \mathcal{F}]$) skeleton obtained by Skolemisation of all the sentences in Φ, where different sets of Skolem symbols are used for different sentences.

The following result is an easy corollary of what we have derived above.

Corollary 1. *For every* $\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]$ *skeleton* $\eth$*, it holds that* $\varphi_\eth$ *is satisfiable* iff $\mathsf{skm}(\eth)$ *is single-time behaviourally satisfiable by a tree* CGS.

Indeed, Theorem 2 ensures that, for every $\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]$ skeleton $\eth$, the sentence $\varphi_\eth$ is satisfiable *iff* $\eth$ is single-time satisfiable by some tree CGS $\mathfrak{G}$. Now, by Theorem 3, $\mathfrak{G}, v \models \phi$ *iff* $\mathfrak{G}, v \models \mathsf{skm}(\phi)$, for all $\phi \in \Phi$ and $v \in \mathrm{Ps}$ with $\ell(\phi) \in \lambda(v)$. Finally, Theorem 4 grants the behavioural satisfaction stated in the corollary.

Any binding prefix $\flat = (a_1, t_1) \cdots (a_k, t_k) \in \mathrm{Bn}(\mathcal{F})$ is a sequence of agent-term pairs, one for each agent. Hence, the standard notions of term replacement, interpretation, unification, and equalisation can be lifted to them in the obvious way. Specifically, $\flat^\mu \triangleq (a_1, t_1^\mu) \cdots (a_k, t_k^\mu)$ denotes the *replacement* of all the variables in every t_i with the terms prescribed by the substitution μ, while $\flat^{\mathfrak{F},\chi}$ denotes the *interpretation* of $\flat$ in $\mathfrak{F}$ under the assignment χ, *i.e.*, the profile $\flat^{\mathfrak{F},\chi} \in \mathrm{Prf}(v)$ assigning to each agent a_i the strategy $t_i^{\mathfrak{F},\chi}$.

A set of binding prefixes $\mathrm{B} \subseteq \mathrm{Bn}(\mathcal{F})$ *unifies* if there is a substitution μ such that $\flat_1^\mu = \flat_2^\mu$, for all $\flat_1, \flat_2 \in \mathrm{B}$, while B *equalises* in $\mathfrak{F}$ if there is an assignment χ such that $\flat_1^{\mathfrak{F},\chi} = \flat_2^{\mathfrak{F},\chi}$, for all $\flat_1, \flat_2 \in \mathrm{B}$.

Example 4. Consider the three binding prefixes $\flat_1 \triangleq (a, x)(b, f_1(x))(c, z)$, $\flat_2 \triangleq (a, f_2)(b, y)(c, y)$, and $\flat_3 \triangleq (a, f_3(x))(b, x)(c, f_3(x))$, obtained by Skolemisation in Example 3. One can see that the two binding prefixes $\flat_1$ and $\flat_2$ unify to

$(a, f_2)(b, f_1(f_2))(c, f_1(f_2))$, while neither $\flat_1$ and $\flat_3$ nor $\flat_2$ and $\flat_3$ unify. By a result in [39] (see Theorem 1) $\flat_1$ and $\flat_2$ also equalise in every structure $\mathfrak{F}$, while there exists a structure $\mathfrak{F}^\star$ (quasi-Herbrand *w.r.t.* $\{\flat_1, \flat_2, \flat_3\}$, see Theorem 2 of the same article) in which $\flat_3$ does not equalise with either $\flat_1$ or $\flat_2$. □

Every finite set of binding prefixes $\mathrm{B} \subset \mathrm{Bn}(\mathcal{F})$ is associated with its *maximally unifiable coverage* $\mathsf{muc}(\mathrm{B}) \subseteq 2^{\mathrm{B}}$, *i.e.*, the unique set of subsets of B such that

1) $\bigcup \mathsf{muc}(\mathrm{B}) = \mathrm{B}$ and
2) every $\mathrm{C} \in \mathsf{muc}(\mathrm{B})$ is maximally unifiable, *i.e.*, C is unifiable, but $\mathrm{C} \cup \{\flat\}$ is not unifiable, for all $\flat \in \mathrm{B} \setminus \mathrm{C}$.

Example 5. Consider the set of three binding prefixes $\mathrm{B} \triangleq \{\flat_4, \flat_5, \flat_6\}$, where $\flat_4 = (\alpha, u)(\beta, v)(\gamma, u)$, $\flat_5 = (\alpha, w)(\beta, f(w))(\gamma, x)$, and $\flat_6 = (\alpha, y)(\beta, z)(\gamma, g(z))$. Then, $\mathsf{muc}(\mathrm{B})$ contains all the subsets of B of size 2. Indeed, the first two binding prefixes unify in $\flat_{45} \triangleq (\alpha, u)(\beta, f(u))(\gamma, u)$, the first and the last unify in $\flat_{46} \triangleq (\alpha, g(v))(\beta, v)(\gamma, g(v))$, and the last two binding prefixes unify in $\flat_{56} \triangleq (\alpha, w)(\beta, f(w))(\gamma, g(f(w)))$. In addition, the full set B is not unifiable, as w cannot unify with $g(f(w))$ and, therefore, $\flat_4$ does not unify with $\flat_{56}$ either. As another example, for the set of binding prefixes $\{\flat_1, \flat_2, \flat_3\}$ of Example 4, we have that $\mathsf{muc}(\{\flat_1, \flat_2, \flat_3\}) = \{\{\flat_1, \flat_2\}, \{\flat_3\}\}$. □

4.3. Normal models

A *normal model* of a universal skeleton $\eth$ is an interpreted tree CGS $(\mathfrak{G}, \Im)$, where the number $|\tau(v)|$ of successors of each position $v \in \mathrm{Ps}$ is dictated solely by the set of binding prefixes $\flat$ of some sentence $\phi \in \Phi$, whose induced play $\pi = \mathsf{play}(\flat^{\Im(w),\chi}, w)$, with $\chi \in \mathsf{Asg}(w)$ and w an ancestor of v satisfying ϕ, passes through v. In other words, each position in a normal model has just enough successors to separate the sets of non-unifying binding prefixes, which may require different paths to satisfy the associated sentences.

The underlying idea is the following. Consider a model of a universal skeleton and a position v in the model labelled with propositions s_1 and s_2, which name the subsentences $\forall \flat_1 \psi_1$ and $\forall \flat_2 \psi_2$, respectively. This witnesses that both sentences must be satisfied at v. If binding prefixes $\flat_1$ and $\flat_2$ unify, hence equalise, then the corresponding LTL matrices[2] ψ_1 and ψ_2 must necessarily be satisfied along a same path from v, as the two binding prefixes induce at least one common path. If, however, $\flat_1$ and $\flat_2$ do not unify, then ψ_1 and ψ_2 can be satisfied independently along different paths, since the binding prefixes are allowed to have different interpretations.

Normal models capture this intuition by keeping track, for each position, of which binding prefix is paired with which path from that position. To keep track of this information, normal models are equipped with three functions:

(i) a *global binding* function g that associates with each position v the set of binding prefixes paired with all the paths through v;
(ii) a *local binding* function l, associating with each position v the set of binding prefixes of the sentences that label v, *i.e.* the sentences that must be satisfied starting from v;
(iii) a *routing* map r that, based on (non)unification of the binding prefixes at v, dispatches them along possibly different paths from v. The notion is formally captured by the following definition.

Definition 7 *(Normal model).* An interpreted CGS $(\mathfrak{G}, \Im)$ satisfying a $\forall\mathrm{SL}^{\emptyset}[1\mathrm{G}]$ skeleton $\eth$ is *normal* if

1. $\mathfrak{G}$ is a tree and
2. there exist three maps $\mathsf{l}, \mathsf{g} \colon \mathrm{Ps} \to 2^{\mathrm{Bn}}$ and $\mathsf{r} \colon v \in \mathrm{Ps} \mapsto (\tau(v) \to 2^{\mathrm{Bn}})$ enjoying the following properties, for all positions $v \in \mathrm{Ps}$:

   a) $\mathsf{l}(v) = \{\flat \in \mathrm{Bn} \mid \forall \flat \psi \in \Phi \text{ and } \ell(\forall \flat \psi) \in \lambda(v)\}$;
   b) $\mathsf{r}(v)$ is a bijective map from $\tau(v)$ to $\mathsf{muc}(\mathsf{g}(v))$;
   c) if $v = \varepsilon$ then $\mathsf{g}(v) = \mathsf{l}(v)$ else $\mathsf{g}(v) = \mathsf{l}(v) \cup \mathsf{r}(\tau^{-1}(v))(v)$;
   d) $\flat \in \mathsf{r}((\pi)_i)((\pi)_{i+1})$, for all $\flat \in \mathsf{l}(v)$, $\chi \in \mathsf{Asg}(v, \mathsf{vr}(\flat))$, and $i \in \mathbb{N}$, where $\pi \triangleq \mathsf{play}(\flat^{\Im(v),\chi}, v)$.

We say that the CGS $(\mathfrak{G}, \Im)$ *normally satisfies* $\eth$. Moreover, we say that $\eth$ is *normally satisfiable* if there exists a CGS $(\mathfrak{G}, \Im)$ that normally satisfies it.

Let us consider the conditions 2a–2d one by one. For each position v, Item 2a ensures that the local binding function l associates with v the set of binding prefixes of every universal sentence $\phi \in \Phi$ whose name $\ell(\phi)$ labels v (note that ϕ holds at v due to Item 2 of Definition 4). Items 2b and 2c are mutually connected. Item 2b asks that each successor position of

[2] Recall that the matrix of a quantified formula φ in prenex form is the maximal quantifier-free subformula of φ.

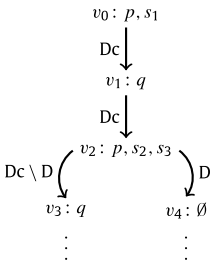

Fig. 1. A normal model with $\mathrm{Dc} \triangleq \{0,1,2\}^{\{a,b,c\}}$ and $\mathrm{D} \triangleq \{(1,0,1),(2,1,2),(1,2,1)\}$.

v be uniquely associated with a maximally unifiable set of binding prefixes attached to v. This ensures that all maximally unifiable binding prefixes are routed towards the same successor and different unifying sets are routed towards different successors. Item 2c, instead, requires that the global binding function g extends l with the binding prefixes of the sentences satisfied at some ancestor of v. Finally, Item 2d captures the requirement that normal models keep track at each position of the coupling between binding prefixes and paths from that position. Indeed, it requires that a path induced by a binding prefix $\flat$ necessarily passes through one of the successors chosen for $\flat$ by r and that, *vice versa*, a successor chosen for $\flat$ is traversed by at least one path induced by $\flat$.

Example 6. Consider the CGS $\mathfrak{G}$ depicted in Fig. 1 over the set of actions $\mathrm{Ac} = \{0,1,2\}$, where the descendants of v_3 (*resp.*, v_4) form a single path of positions with the same labelling of v_3 (*resp.*, v_4) and the set Dc denotes all the possible triples of decisions.

The sentence φ of Example 1 admits the following Skolem skeleton:

$$\eth = \langle p \wedge s_1, \{\phi_1, \phi_2, \phi_3\}, \{\phi_1 \mapsto s_1, \phi_2 \mapsto s_2, \phi_3 \mapsto s_3\}\rangle, \text{ where}$$

$$\begin{aligned} &\phi_1 \triangleq \forall x \forall z \flat_1 (\mathtt{X}\,(q \wedge (\mathtt{X}\,\mathtt{F}\,q)\,\mathtt{U}\,(s_2 \wedge s_3))), && \flat_1 \triangleq (a,x)(b,f_1(x))(c,z),\\ &\phi_2 \triangleq \forall y \flat_2 (p\,\mathtt{U}\,q), && \flat_2 \triangleq (a,f_2)(b,y)(c,y),\\ &\phi_3 \triangleq \forall x \flat_3 (\mathtt{G}\,\neg q), && \flat_3 \triangleq (a,f_3(x))(b,x)(c,f_3(x)). \end{aligned} \quad \textit{and}$$

Let us take a strategy interpretation $\Im$ defined as follows:

(i) $f_1^{\Im(v_0)} \colon \mathrm{Str}(v_0) \to \mathrm{Str}(v_0)$ is the identity function, *i.e.*, $f_1^{\Im(v_0)}(\sigma) = \sigma$;
(ii) $f_2^{\Im(v_2)} \in \mathrm{Str}(v_2)$ is the constant strategy such that $f_2^{\Im(v_2)}(\rho) = 0$, for all histories $\rho \in \mathrm{Hst}(v_2)$;
(iii) $f_3^{\Im(v_2)} \colon \mathrm{Str}(v_2) \to \mathrm{Str}(v_2)$ is the function mapping any strategy $\sigma \in \mathrm{Str}(v_2)$ to some strategy $f_3^{\Im(v_2)}(\sigma)$ that, when applied to the one-length history v_2, satisfies the equality $f_3^{\Im(v_2)}(\sigma)(v_2) = 1 + (\sigma(v_2) \bmod 2)$.

It can be shown that the interpreted CGS $(\mathfrak{G}, \Im)$ is a normal model for $\eth$. Consider the following three maps l, g, and r:

(a) $\mathsf{l}(v_0) = \{\flat_1\}$, $\mathsf{l}(v_2) = \{\flat_2, \flat_3\}$, and $\mathsf{l}(v) = \emptyset$ for all the other positions v in the CGS $\mathfrak{G}$;
(b) $\mathsf{g}(v_0) = \mathsf{g}(v_1) = \{\flat_1\}$, $\mathsf{g}(v_2) = \{\flat_1, \flat_2, \flat_3\}$, $\mathsf{g}(v) = \{\flat_1, \flat_2\}$, for all $v \in \tau^*(v_3)$, and $\mathsf{g}(v) = \{\flat_3\}$, for all $v \in \tau^*(v_4)$;
(c) $\mathsf{r}(v_0)(v_1) = \mathsf{r}(v_1)(v_2) = \{\flat_1\}$, $\mathsf{r}(v_2)(v_3) = \{\flat_1, \flat_2\} = \mathsf{r}(v)(v')$, with $v' \in \tau(v)$ and $v \in \tau^*(v_3)$, and $\mathsf{r}(v_2)(v_4) = \{\flat_3\} = \mathsf{r}(v)(v')$, with $v' \in \tau(v)$ and $v \in \tau^*(v_4)$.

The function l clearly satisfies Item 2a. Indeed, only v_0 and v_2 are labelled with names of subsentences and the corresponding binding prefixes are associated with them by the function l. Specifically, $s_1 = \ell(\phi_1) \in \lambda(v_0)$ and $\mathsf{l}(v_0) = \{\flat_1\}$, while $\{s_2, s_3\} = \{\ell(\phi_2), \ell(\phi_3)\} \subseteq \lambda(v_3)$ and $\mathsf{l}(v_3) = \{\flat_2, \flat_3\}$. Bijectivity of r on each position v is immediate, since $\mathsf{r}(v)$ assigns to each outgoing move exactly one of the sets of binding prefixes in $\mathsf{muc}(\mathsf{g}(v))$ as given in Example 4, hence Item 2b holds. For each position v except v_0, $\mathsf{g}(v)$ collects the binding prefixes in $\mathsf{l}(v)$ and those assigned by r to its incoming moves, hence we have Item 2c. Finally, to check Item 2d, we only need to consider the plays out of v_0 and v_2, as these are the only positions to which l assigns some binding prefixes. First observe that, given the strategy interpretation $\Im$, the only play from v_0 induced by $\flat_1$ is $v_0 v_1 v_2 v_3 \cdots$. Indeed, $\flat_1 = (a,x)(b,f_1(x))(c,z)$ and $f_1^{\Im(v_0)}$ is the identity, hence the binding prefix only allows for choices of the form (k,k,j) (with $k, j \in \{0,1,2\}$), where agents a and b always choose the same action. This forces the plays to turn left at position v_2 towards v_3. Since $\flat_1$ is associated with all the moves in those plays, Item 2d holds at v_0. Let us consider position v_2 next, whose associated binding prefixes $\flat_2$ and $\flat_3$ do not unify as observed in Example 4. Since $f_2^{\Im(v_2)}$ is the constant strategy that chooses 0 everywhere, the only admissible decisions compatible with $\flat_2$ at v_2 are of the form $(0,k,k)$ (with $k \in \{0,1,2\}$). Therefore, the only compatible play from v_2 takes (v_2, v_3) as the first move and $\flat_2$ is routed along that move by r. As to $\flat_3$, it is not hard to check that the compatible decisions at v_2 are precisely the ones in the set D. Indeed, whatever value is assigned to x by an assignment χ, the response of $f_3^{\Im(v_2)}(\chi(x))$ to the one-length

history v_2 would be either 1 or 2. Hence, the play compatible with $\flat_3$ takes (v_2, v_4) as first move and $\flat_3$ is routed along that move by r. This shows that Item 2d holds at v_2 as well.

Now it is immediate to verify that the model satisfies the skeleton according to Definition 4. The sentence ϕ_1 is satisfied at v_0 since the only play compatible with $\flat_1$, namely $v_0 v_1 v_2 v_3 \cdots$, satisfies the matrix $(\mathtt{X}(q \wedge (\mathtt{X}\,\mathtt{F}\,q)\,\mathtt{U}\,(s_2 \wedge s_3)))$. The sentence ϕ_2 is satisfied at v_2, as the play compatible with $\flat_2$, namely $v_2 v_3 \cdots$, satisfies $p\,\mathtt{U}\,q$. Finally, the sentence ϕ_3 is satisfied at v_2, since the play compatible with $\flat_3$, namely $v_2 v_4 \cdots$, satisfies $\mathtt{G}\neg q$. Single-time satisfaction is ensured by construction, since each name for a subsentence in the skeleton is present in the labelling of a single position, in accordance with Definition 6. □

The main result of the section is that the Skolem skeleton $\mathsf{skm}(\eth)$ of any satisfiable $\mathrm{SL}^{\mathcal{O}}[1\mathrm{G}]$ skeleton $\eth$ is satisfied by a normal model. By Corollary 1, $\mathsf{skm}(\eth)$ is single-time behaviourally satisfied by an interpreted tree CGS $(\mathfrak{G}, \Im)$. The idea is to build the normal model starting from $(\mathfrak{G}, \Im)$. First, it is easy to convince oneself that there exist three functions $\mathsf{l}, \mathsf{g} \colon \mathrm{Ps} \to 2^{\mathrm{Bn}}$ and $\mathsf{r} \colon v \in \mathrm{Ps} \mapsto (\tau(v) \to 2^{\mathrm{Bn}})$ satisfying Items 2a, 2c, and 2d of Definition 7, since these items can actually be interpreted as definitions. However, Item 2b may not hold, as $\mathsf{r}(v)(w)$ may not belong to $\mathsf{muc}(\mathsf{g}(v))$ or $\mathsf{r}(v)$ may not be a bijection from $\tau(v)$ to $\mathsf{muc}(\mathsf{g}(v))$. On the one hand, the move relation of $\mathfrak{G}$ may force $\mathsf{r}(v)$ to route non-unifying binding prefixes along the same move. On the other hand, two unifying binding prefixes may be routed along two different moves. In both cases the bijection requirement would fail to hold.

The idea is, then, to build a new tree model by suitably changing the move relation of $\mathfrak{G}$ in such a way that r has exactly the right number of moves to route unifying binding prefixes in a bijective manner. Thanks to the behaviouralness property of $\Im$, this can be done position-wise, *e.g.*, by following a breadth-first ordering of the nodes of the tree. Indeed, one can decompose stepwise the strategy interpretation $\Im(v) = \langle \mathrm{Str}(v), \cdot^{\Im(v)} \rangle$ at each position $v \in \mathrm{Ps}$ into infinitely-many function interpretations $\mathfrak{F}^v_\rho = \langle \mathrm{Ac}, \cdot^{\mathfrak{F}^v_\rho} \rangle$, called *action interpretations*, one for each possible history $\rho \in \mathrm{Hst}(v)$ starting at v. Each such action interpretation provides a first-order snapshot of the strategies in $\Im(v)$ at a given history and its domain is the set of actions in their range. Specifically, for each function symbol $f \in \mathrm{Fn}$ of arity $k \in \mathbb{N}$, we can set

$$f^{\mathfrak{F}^v_\rho}(\vec{c}) \triangleq f^{\Im(v)}(\vec{\sigma})(\rho),$$

for all k-tuples of strategies $\vec{\sigma} \in \mathrm{Str}(v)^k$, where the i-th element $(\vec{c})_i$ of $\vec{c} \in \mathrm{Ac}^k$ is equal to the action $(\vec{\sigma})_i(\rho)$ chosen by the i-th strategy $(\vec{\sigma})_i$ of $\vec{\sigma}$ at ρ. Basically, each action interpretation $\mathfrak{F}^v_\rho$ encodes the responses that the strategies in $\Im(v)$ give against the history ρ. Then, the strategy interpretation $\Im(v)$ can be viewed as a tree of action interpretations $\mathfrak{F}^v_{\rho_w}$, one for each descendant w of v, where ρ_w is the history starting in v and leading to w. Observe that for this stepwise decomposition to be faithful, behaviouralness of $\Im(v)$ is crucial, since the action interpretation $\mathfrak{F}^v_\rho$ only contains information on what the strategies dictate on ρ. A non-behavioural $\Im(v)$, by contrast, may choose responses at history ρ that look at the choices that the strategies make in the future of ρ and this cannot be accounted for in the action interpretation $\mathfrak{F}^v_\rho$.

Thanks to the results in [39] reported below, given the set of binding prefixes of the skeleton, we can build a first-order interpretation $\mathcal{H} = \langle \mathrm{H}, \cdot^{\mathcal{H}} \rangle$ on which all sets of binding prefixes unify *iff* they equalise in $\mathcal{H}$. Recall that a set Z of binding prefixes equalises in $\mathcal{H}$ if there exist two first-order assignments $\xi_{\mathrm{Vr}} \colon \mathrm{Vr} \to \mathrm{H}$ and $\xi_{\mathrm{Ag}} \colon \mathrm{Ag} \to \mathrm{H}$ such that, when each variable is assigned to the value prescribed by the assignment ξ_{Vr}, the interpretations of all binding prefixes in Z coincide with ξ_{Ag}, in symbols $\flat^{\mathcal{H}, \xi_{\mathrm{Vr}}} = \xi_{\mathrm{Ag}}$, for all $\flat \in \mathrm{Z}$. The correspondence between unification and equalisation *w.r.t.* a set T of terms in a first-order structure is called *quasi-Herbrand w.r.t.* T in [39]. Observe that, when the domain H is interpreted as the set of actions of a CGS, the assignment ξ_{Ag} is indeed a decision of that CGS, *i.e.*, the binding prefixes in Z equalise on a decision. The first-order interpretation $\mathcal{H}$ allows us to obtain, for each position v, the maximally unifiable coverage $\mathsf{muc}(\mathsf{g}(v))$ of the set of binding prefixes at v. The new model is, then, defined so that it has precisely one move out of v for each element of $\mathsf{muc}(\mathsf{g}(v))$ and along each such move only one maximal set of unifying binding prefixes in $\mathsf{muc}(\mathsf{g}(v))$ will be routed by r. Once this operation is performed iteratively on all the positions of the original tree, the required bijectivity property will be enforced.

The objective of the above transformation is twofold. On the one hand, *(i)* it removes redundant branches in the model that satisfy goals whose binding prefixes equalise and, thus, can be satisfied by the unique branch that is preserved in the normal model. On the other hand, *(ii)* it separates binding prefixes that do not equalise and whose goals can be satisfied independently. Here, satisfaction is preserved by adding in the normal model a distinct copy of the original satisfying branch, one for each separated (and non-equalising) binding prefix. As mentioned above, the result relies on an earlier result that, for convenience, we reformulate here in the context of the present work.

Theorem 5 *(Theorem 1 in [39]). For every set* $\mathrm{T} \subseteq \mathrm{Tr}$ *of* $\mathcal{F}$*-terms and every* $\mathcal{F}$*-structure* $\mathfrak{F}$*, if* T *unifies then* T *equalises in* $\mathfrak{F}$*.*

In our context, this ensures that for every unifiable set $\mathrm{B} \subseteq \mathrm{Bn}$ of binding prefixes there exists a decision $d \in \mathrm{Dc}$ on which B equalises.

Theorem 6 *(Theorem 2 in [39]). For every finite set* $\mathrm{T} \subseteq \mathrm{Tr}$ *of terms over* $\mathcal{F}$*, there exists a finite* $\mathcal{F}$*-structure which is quasi-Herbrand* w.r.t. T.

The quasi-Herbrand property stated in the theorem, when applied to the set of terms occurring in a set $\mathrm{B} \subseteq \mathrm{Bn}(\mathcal{F})$ of binding prefixes, corresponds to the existence of an $\mathcal{F}$-structure $\mathcal{H}$ such that every $\mathrm{X} \subseteq \mathrm{B}$ equalises in $\mathcal{H}$ *iff* X unifies, which is precisely what we are looking for.

Theorem 7. *For every* $\mathrm{SL}^{\not\ni}\,[1\mathrm{G}]$ *skeleton* $\eth$*, it holds that* $\varphi_\eth$ *is satisfiable* iff $\mathsf{skm}(\eth)$ *is single-time normally satisfiable.*

Proof. The *'if'* direction of the theorem is trivial, so let us focus on the *'only if'* one. Assume $\varphi_\eth$ is satisfiable. By Corollary 1, there is an interpreted tree CGS $(\mathfrak{G}, \Im)$ that single-time behaviourally satisfies the universal skeleton $\mathsf{skm}(\eth)$. As observed above, for every position $v \in \mathrm{Ps}$ and history $\rho \in \mathrm{Hst}(v)$, there is an action interpretation $\mathfrak{F}^v_\rho$ corresponding to the first-order snapshot of the strategy interpretation $\Im(v)$ at ρ. Actually, thanks to the following claim, we can choose the interpreted tree CGS $(\mathfrak{G}, \Im)$ in such a way that, for every position $v \in \mathrm{Ps}$, there exists a unique action interpretation $\mathfrak{F}_v$ equal to $\mathfrak{F}^u_{\rho v}$, for all positions $u \in \mathrm{Ps}$ and histories $\rho v \in \mathrm{Hst}(u)$.

Claim 1. *For every* $\mathrm{SL}^{\not\ni}\,[1\mathrm{G}]$ *skeleton* $\eth$*, it holds that* $\varphi_\eth$ *is satisfiable* iff *there exist a tree* CGS $\mathfrak{G}$ *and a function interpretation* $\Im$ *enjoying the following:*

1. *the interpreted* CGS $(\mathfrak{G}, \Im)$ *single-time behaviourally satisfies* $\mathsf{skm}(\eth)$*;*
2. *for all function symbols* $f \in \mathrm{Fn}$ *of some arity* $k \in \mathbb{N}$*, positions* $v, u \in \mathrm{Ps}$*, histories* $\rho v \in \mathrm{Hst}(u)$*, and* k*-vectors of strategies* $\vec{\sigma} \in \mathrm{Str}(v)^k$ *and* $\vec{\sigma}\,' \in \mathrm{Str}(u)^k$ *with* $(\vec{\sigma})_i(v) = (\vec{\sigma}\,')_i(\rho v)$*, for all* $0 \leq i < k$*, it holds that* $f^{\Im(v)}(\vec{\sigma})(v) = f^{\Im(u)}(\vec{\sigma}\,')(\rho v)$*.*

The truth of this statement immediately follows from the fact that, due to the single-time satisfaction, once a universal sentence $\forall \flat \psi \in \Phi$ is satisfied at a position u, it will never be required to hold at any descendant v along a history ρv rooted in u. Thus, we can replace the strategy interpretation at v of any function f used in the binding prefix $\flat$ with the one given at u on the history ρv. Obviously, such a change does not interfere with the satisfaction of the universal sentence $\forall \flat \psi$ at u.

To prove the theorem, let v be the first position of $\mathfrak{G}$ in a breadth-first order that does not satisfy Item 2b of Definition 7. We now show how to modify $\mathfrak{G}$ to correct this problem.

Let $\mathrm{B} \triangleq \mathrm{g}(v)$ be the set of binding prefixes routed through v. By Theorem 6, there exists a first-order structure $\mathcal{H}$ over some finite set H such that, for any subset $\mathrm{Z} \subseteq \mathrm{B}$, it holds that Z equalises in $\mathcal{H}$ *iff* it unifies.

Before proceeding, observe that, for every set X with $|\mathrm{X}| \geq |\mathrm{Ac}|$, where Ac is the set of actions of $(\mathfrak{G}, \Im)$, there exists an interpreted tree CGS with X as the set of actions that still satisfies the two items of the above claim. Indeed, one can exploit an arbitrary surjective map $f : \mathrm{X} \to \mathrm{Ac}$ to construct such a model in the obvious way: every action c in X behaves as the action $f(c)$ in Ac. Similarly, for every set X with $|\mathrm{X}| \geq |\mathrm{H}|$, where H is the domain of $\mathcal{H}$, there exists a quasi-Herbrand model for B with X as the domain. Therefore, *w.l.o.g.*, we assume that the set of actions of $(\mathfrak{G}, \Im)$ and the domain H of $\mathcal{H}$ coincide.

Now, for each decision $d \in \mathrm{Dc}$, let $\mathrm{U}_d \subseteq \mathrm{g}(v)$ be the set of binding prefixes at position v that equalise on d in the quasi-Herbrand model $\mathcal{H}$, *i.e.*,

$$\mathrm{U}_d \triangleq \left\{ \flat \in \mathrm{g}(v) \;\middle|\; \exists \xi : \mathrm{Ac}^{\mathrm{Vr}} \to \mathrm{Ac}.\, \flat^{\mathcal{H},\xi} = d \right\}.$$

In addition, let $\mathscr{U} \triangleq \{\mathrm{U}_d \mid d \in \mathrm{Dc}\}$ and $\Gamma : \mathscr{U} \to \mathsf{muc}(\mathrm{g}(v))$ be a choice function associating each set U_d with a maximal unifying set $\Gamma(\mathrm{U}_d)$ containing it, *i.e.*, such that $\mathrm{U}_d \subseteq \Gamma(\mathrm{U}_d)$.

At this point, we construct a new interpreted tree CGS $(\widehat{\mathfrak{G}}, \widehat{\Im})$ over the same set of actions Ac of $(\mathfrak{G}, \Im)$ that still single-time behaviourally satisfies the universal skeleton $\mathsf{skm}(\eth)$. We define the tree CGS $\widehat{\mathfrak{G}}$ and the action interpretations $\widehat{\mathfrak{F}}_v$, one for each position v in $\widehat{\mathfrak{G}}$.

- For all positions $u \in \mathrm{Ps}$ which is not a descendant of v, we have (a) $u \in \widehat{\mathrm{Ps}}$, (b) $\widehat{\tau}(u, d) \triangleq \tau(u, d)$, for all decisions $d \in \mathrm{Dc}$, (c) $\widehat{\lambda}(u) \triangleq \lambda(u)$, and (d) $\widehat{\mathfrak{F}}_u \triangleq \mathfrak{F}_u$.
- Let $\{v_{\mathrm{U}} \mid \mathrm{U} \in \mathsf{muc}(\mathrm{g}(v))\}$ be a fresh set of $|\mathsf{muc}(\mathrm{g}(v))|$ positions that contains the successors of v in $\widehat{\mathfrak{G}}$, one for each maximally unifiable subset of binding prefixes in $\mathrm{g}(v)$. We then set (a) $\widehat{\tau}(v, d) \triangleq v_{\Gamma(\mathrm{U}_d)}$, for all decisions $d \in \mathrm{Dc}$, (b) $\widehat{\lambda}(v) \triangleq \lambda(v)$, and (c) $\widehat{\mathfrak{F}}_v \triangleq \mathcal{H}$.
- Every successor v_{U} of v in $\widehat{\mathfrak{G}}$ (recall that $\mathrm{U} \in \mathsf{muc}(\mathrm{g}(v))$) is the root of a subtree which is an isomorphic copy, in both the structure and the labelling, of the subtree in the original CGS $\mathfrak{G}$ rooted in any position $v_d = \tau(v, d)$ such that $\mathrm{U}_d = \mathrm{U}$. Observe that such a decision d must exist thanks to Theorem 5. Let ι be a tree isomorphism mapping the subtree of $\widehat{\mathfrak{G}}$ rooted in v_{U} to the subtree of $\mathfrak{G}$ rooted in v_d. For all u descendants of v_{U} in $\widehat{\mathfrak{G}}$, we then set (a) $\widehat{\tau}(u, d) \triangleq \tau(\iota(u), d)$, for all decisions $d \in \mathrm{Dc}$, (b) $\widehat{\lambda}(u) \triangleq \lambda(\iota(u))$, and (c) $\widehat{\mathfrak{F}}_u \triangleq \mathfrak{F}_{\iota(u)}$.

The new behavioural strategy interpretation $\widehat{\Im}$ can immediately be obtained by recomposing the first-order snapshots given by the action interpretations $\{\widehat{\mathfrak{F}}_v\}_{v \in \widehat{\mathrm{Ps}}}$, via the correspondence between the two functional structures, already outlined before Theorem 5. Formally, for each position $u \in \widehat{\mathrm{Ps}}$ and function symbol $f \in \mathrm{Fn}$ of arity $k \in \mathbb{N}$, we can set

$$f^{\widehat{\mathfrak{I}}(u)}(\vec{\sigma})(\rho v) \triangleq f^{\widehat{\mathfrak{F}}_v}(\vec{c}),$$

for all k-tuples of strategies $\vec{\sigma} \in \mathrm{Str}(u)^k$, histories $\rho v \in \mathrm{Hst}(u)$, and tuples of actions $\vec{c} \in \mathrm{Ac}^k$ whose i-th element $(\vec{c})_i$ is equal to the action $(\vec{\sigma})_i(\rho v)$ chosen by the i-th strategy $(\vec{\sigma})_i$ of $\vec{\sigma}$ at ρv.

Now, let $\widehat{\mathsf{l}}, \widehat{\mathsf{g}}\colon \widehat{\mathrm{Ps}} \to 2^{\mathrm{Bn}}$ and $\widehat{\mathsf{r}}\colon v \in \widehat{\mathrm{Ps}} \mapsto (\tau(v) \to 2^{\mathrm{Bn}})$ be the three functions for $(\widehat{\mathfrak{G}}, \widehat{\mathfrak{I}})$ satisfying Items 2a, 2c, and 2d of Definition 7. By construction, we have that $\widehat{\mathsf{r}}(\varepsilon)(v_{\mathrm{U}}) = \mathrm{U} \in \mathsf{muc}(\mathsf{g}(\varepsilon))$, which means that $\widehat{\mathsf{r}}(\varepsilon)$ is bijective. Thus, Item 2b is satisfied at position v, as a result of the patching applied to it.

By applying the above transformation to all other positions in a breadth-first manner, we obtain the required model for the skeleton $\mathsf{skm}(\eth)$. Indeed, assume by contradiction that the interpreted tree CGS $(\mathfrak{G}^*, \mathfrak{I}^*)$ obtained at the limit does satisfy the invariant (which is preserved by every individual correction step), *i.e.*, that it does not single-time behaviourally satisfies the universal skeleton $\mathsf{skm}(\eth)$. Obviously, the function interpretation $\mathfrak{I}^*$ is behavioural by construction, since it is built by recomposing the local first-order snapshots. Thus, the only possibility for violating the invariant is the existence of a position v in $\mathfrak{G}^*$ labelled with some atomic proposition $\ell(\wp\flat\psi)$, with $\wp\flat\psi \in \Phi$, such that $(\mathfrak{G}^*, \mathfrak{I}^*), v \not\models \mathsf{skm}(\wp\flat\psi)$. This means that there exists an assignment $\chi \in \mathrm{Asg}(v, \mathsf{vr}(\flat'))$, such that $(\mathfrak{G}^*, \mathfrak{I}^*), (v, \flat'^{\mathfrak{I}(v),\chi}) \not\models \psi$, where $\flat'$ is the binding of $\mathsf{skm}(\wp\flat\psi)$. Since ψ is an LTL formula, its satisfaction only depends on the labelling of the play $\mathsf{play}(\flat^{\mathfrak{I}(v),\chi}, v)$ induced by χ. However, by the correction procedure described above, such a labelling coincides with the one of some play π starting from v in the original model $\mathfrak{G}$ and that is induced by some assignment of the universal variables in $\flat'$. As a consequence, π would not satisfy ψ either, contradicting the fact that $(\mathfrak{G}, \mathfrak{I})$ was a model of the universal skeleton $\mathsf{skm}(\eth)$ to begin with. □

The main result of this section states that every satisfiable $\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]$ sentence has a bounded-fork model. This can be easily derived from the previous theorem by observing the following:

(1) due to the single-time satisfaction property, along any path of the model, there are at most $|\Phi|$ sentences of the form $\mathsf{skm}(\wp\flat\psi)$ that need to be satisfied, since every atomic proposition $\ell(\wp\flat\psi)$ occurs at most once;
(2) thanks to the normality property, a fork at any given position v of a path is only caused by non-unifying binding prefixes, which occur if new sentences in Φ need to be satisfied at v, as the binding prefixes routed toward v from the ancestors necessarily unify.

Corollary 2. *For every* $\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]$ *skeleton* $\eth$*, it holds that* $\varphi_\eth$ *is satisfiable* iff $\mathsf{skm}(\eth)$ *is single-time normally satisfied by a* k*-fork* CGS*, for* $k \triangleq \max\{0, |\Phi| - 1\}$.

Proof. Let us consider the case where $|\Phi| > 0$, so, $k = |\Phi| - 1$, the other case being trivial. Suppose that $\varphi_\eth$ is satisfiable and let $\mathfrak{G}$ be the CGS single-time normally satisfying the universal skeleton $\mathsf{skm}(\eth)$, whose existence is ensured by Theorem 7. We necessarily have that $\mathfrak{G}$ is a k-bounded-fork tree. Indeed, suppose by contradiction that this does not hold. Then, there exists a path $\pi \in \mathrm{Pth}(v_I)$ and $k+1$ indices $i_1 < \ldots < i_{k+1}$ such that $\left|\tau((\pi)_{i_j})\right| > 1$, for all $j \in [1, k+1]$. Since $\mathfrak{G}$ is normal, by Item 2b of Definition 7, it holds that $\left|\mathsf{muc}(\mathsf{g}((\pi)_{i_j}))\right| > 1$. Therefore, by Item 2c, for all $j \in [2, k+1]$, there exists a binding prefix $\flat_j \in \mathsf{l}((\pi)_{i_j})$ not belonging to $\mathsf{r}((\pi)_{i_j-1})((\pi)_{i_j})$, since all binding prefixes in this last set unify. This implies that there exists a sentence $\phi_j \in \Phi$ such that $\ell(\phi_j) = \lambda((\pi)_{i_j})$, for every index $j \in [2, k+1]$, due to Item 2a. For the same reasons, there must exist a sentence $\phi_1 \in \Phi$ such that $\ell(\phi_1) \in \lambda((\pi)_{i_1})$ and a different sentence $\phi_0 \in \Phi$ whose binding prefix belongs to $\mathsf{g}((\pi)_{i_1})$. This means that $\ell(\phi_0) \in \lambda((\pi)_{i_0})$, for some $i_0 \leq i_1$. Since $|\Phi| = k+1$, there exist two indices $0 \leq j_1 < j_2 \leq k+1$ such that $\phi_{j_1} = \phi_{j_2}$, but this contradicts the fact that $\mathfrak{G}$ single-time satisfies $\mathsf{skm}(\eth)$, since no labelling of a sentence in Φ can appear twice along a path of $\mathfrak{G}$. □

5. New classes of automata

We have shown in the previous section that every satisfiable sentence φ of the non-recurrent fragments of SL[1G] is satisfiable by a normal model, which, in turn, only contains along each path a number of forks bounded by the number of subsentences of φ, hence, also by its size. We shall exploit this strong property in Section 6 to efficiently solve the satisfiability problem for such fragments, by reducing it to the emptiness problem of a new class of tree automata, called *bounded-fork tree automata*, that accept k-fork tree CGSs. For the reduction, we shall also leverage a novel kind of good-for-games word automata, called *prefix-deterministic word automata*, that can easily be embedded into tree automata to allow for checking linear properties along all the branches of the input tree.

5.1. Bounded-fork tree automata

Bounded-fork automata are a restriction of the standard tree automata tailored to accept only trees having a bounded number of forks along each path starting from the root, which enjoys a computationally simpler emptiness problem. If at most k forks in a path are allowed, we can endow the automaton with the ability to count the number of occurring forks along the paths by partitioning the set of states Q into $k+1$ subsets $\mathrm{Q}_0, \ldots, \mathrm{Q}_k$. Intuitively, a state $q \in \mathrm{Q}_i$ can observe at

Algorithm 1: k-NBT non-emptiness check.

```
   function nonemp(A)
   |  let ⟨_, _, _, _, q_I, _⟩ = A in
1  |  return nonemp(A, q_I, 0)

   function nonemp(A, q, c)
   |  let ⟨Σ, Λ, ⋃_{j=0}^{k} Q_j, δ, _, Q_F⟩ = A in
1  |  if q ∈ Q_0 then
2  |  |  return irch(G_A, Q_F, q)
   |  else
3  |  |  if c < |⋃_{j=1}^{k} Q_j| then
4  |  |  |  foreach σ ∈ Σ, d ∈ Λ do
5  |  |  |  |  foreach q⃗ ∈ δ(q, σ, d) do
6  |  |  |  |  |  α ← ⊤
7  |  |  |  |  |  foreach i ∈ [1, d] do
8  |  |  |  |  |  |  if ¬nonemp(A, (q⃗)_i, c + 1) then
9  |  |  |  |  |  |  |  α ← ⊥
10 |  |  |  |  |  |  |  break
11 |  |  |  |  |  if α then
12 |  |  |  |  |  |  return ⊤
13 |  |  return ⊥
```

Algorithm 2: Infinite reachability check.

```
   function irch(G, W, v)
1  |  foreach w ∈ W do
2  |  |  if rch(G, v, w) then
3  |  |  |  foreach u ∈ E_G(w) do
4  |  |  |  |  if rch(G, u, w) then
5  |  |  |  |  |  return ⊤
6  |  return ⊥
```

most i additional forks along each path of the input subtree. Naturally, the initial states belong to Q_k and only states in Q_0, from where no more forks are admitted, may be involved in the Büchi acceptance condition.

Definition 8 *(Bounded-fork automaton).* An NTA $\mathcal{A} \triangleq \langle \Sigma, \Lambda, \mathrm{Q}, \delta, q_I, \mathrm{Q_F} \rangle$ is *k-forking* (k-NTA , for short), for some given $k \in \mathbb{N}$, if the following holds true:

1) $1 \in \Lambda$, *i.e.*, 1 is an admissible node degree;
2) there exists an ordered $(k+1)$-partition $(\mathrm{Q}_0, \ldots, \mathrm{Q}_k)$ of Q for which it holds that:
   a) $\delta(q, \sigma, 1) \subseteq \bigcup_{j=0}^{i} \mathrm{Q}_j$ and $\delta(q, \sigma, d) \subseteq (\bigcup_{j=0}^{i-1} \mathrm{Q}_j)^d$, for all indices $i \in [0, k]$, states $q \in \mathrm{Q}_i$, input symbols $\sigma \in \Sigma$, and node degrees $d \in \Lambda \setminus \{1\}$;
   b) $q_I \in \mathrm{Q}_k$;
   c) $\mathrm{Q_F} \subseteq \mathrm{Q}_0$.

The motivation for using k-NTAs, instead of standard NTAs, is clearly expressed by Theorem 8, which establishes a logarithmic space complexity of the emptiness problem *w.r.t.* the size of the k-NTA . This contrasts with the PTime hardness bound on the same problem for classic NTA shown in [54]. To prove the result, we devise a (deterministic) recursive reachability algorithm, whose pseudo-code is reported in Algorithm 1, that looks for a reachable cycle which includes an accepting state and is completely contained within the partition Q_0 of the k-NTA . The gain in complexity is due to the fact that the algorithm only needs the space required for backtracking along a path to previous fork states, whose number is bounded by k. The emptiness problem can also be solved by reduction to alternating Turing machines [16,17], as well as to Büchi games [55], where the number of turns assigned to the universal player is limited by the value k [56].

Theorem 8. *The emptiness problem for a k-NTA with n states and a transition function of size m can be solved in* $\textsc{DSpace}(\Theta(k \cdot \log n + (\log n)^2 + \log m))$ *and* $\textsc{ATime}[k\text{-alt}](\Theta(\log n + \log m))$*, in both case in an on-the-fly fashion.*

Proof. We first show that the emptiness problem of a k-NTA $\mathcal{A} \triangleq \langle \Sigma, \Lambda, Q, \delta, q_I, Q_F \rangle$ can be solved in ATIME[k-alt]($\Theta(\log n + \log m)$), *i.e.*, by means of an alternating Turing machine $\mathcal{M}$ with at most k universal steps and tape space only logarithmic in the size of the automaton. Beside a binary counter maintained on the tape, which is used to identify the existence of an accepting cycle, the state space of the machine coincides with the set of states $S_\exists \triangleq Q$ of the automaton plus the subset $S_\forall \triangleq \bigcup_{q \in Q} \bigcup_{\sigma \in \Sigma} \bigcup_{d \in \Lambda \setminus \{1\}} \delta(q, \sigma, d)$ of the tuples contained in its transition function: the single states in $S_\exists$ are controlled by the existential player underlying the alternating semantics of the machine, while the tuples of states in $S_\forall \subseteq \bigcup_{d \in \Lambda \setminus \{1\}} Q^d$ are controlled by the universal one. When in an existential state $q \in S_\exists$, the machine $\mathcal{M}$ nondeterministically chooses an element from $\bigcup_{\sigma \in \Sigma} \bigcup_{d \in \Lambda} \delta(q, \sigma, d)$. Conversely, when in a universal state $\vec{q} \in S_\forall$, $\mathcal{M}$ universally chooses a direction $i \in [0, |\vec{q}|)$ of the tree and, so, the corresponding i-th coordinate element $(\vec{q})_i \in Q$ in $\vec{q}$. Obviously, $\mathcal{M}$ cannot make more than k universal choices in a single execution run, thanks to the k-partitioning of the state space of the original automaton $\mathcal{A}$. Observe that the Turing Machine need not have access to the entire automaton $\mathcal{A}$ at once, since the definition of transition function for a given (machine) state only requires knowledge of the transitions for the corresponding state of the automaton. Thus, the emptiness check can be solved in an on-the-fly fashion.

Let us now focus on a deterministic algorithm to solve the emptiness problem, which provides us with a procedure with the claimed DSPACE($\Theta(k \cdot \log n + (\log n)^2 + \log m)$) complexity. The pseudo-code of the algorithm is given in Algorithm 1 and relies on the subroutine irch($\mathcal{G}, W, v$), reported in Algorithm 2, that solves the infinite-reachability problem of the vertices in W starting from a given initial vertex v in the graph $\mathcal{G}$. This is done by means of repeated applications of the classic deterministic reachability test implemented by the function rch($\mathcal{G}, v, w$). Intuitively, the main function nonemp($\mathcal{A}$) calls the recursive function nonemp($\mathcal{A}, q, c$) to look for a finite run, *i.e.*, a sequence of states $\vartheta = q_0, q_1, \ldots, q_h$, that, starting from the initial one $q_0 = q_I$, leads to a state $q_h \in Q_0$ in the last partition of the state space of $\mathcal{A}$ and contains the set of Büchi accepting states Q_F. From that state, then, irch($\mathcal{G}_\mathcal{A}, Q_F, q$) looks for some accepting infinite extension ϑ' of ϑ, where $\mathcal{G}_\mathcal{A}$ is the graph underlying the automaton. Since this acceptance condition is memoryless, we do not need to search for non-simple accepting runs. This means that it suffices to look for runs whose prefix contained in $Q \setminus Q_0$ does not exceed the number of states in that set. It is well-known that the reachability problem can be deterministically solved in DSPACE($\Theta\big((\log n)^2\big)$) [57]. As a consequence, the deterministic irch algorithm also requires $\Theta\big((\log n)^2\big)$ space. The rest of the procedure only needs $\Theta(k \cdot \log n + \log m)$ additional space in order to maintain the global counter c, the tuple of states $\vec{q}$, and the k states for which a tuple longer than 1 has been selected from the transition function. Finally, observe that Algorithm 1 can decide the emptiness of the input automaton $\mathcal{A}$ on-the-fly, since it only needs to have access to the portion of the transition function for the current state. □

5.2. Good-for-games automata

One way to solve the satisfiability problem for branching-time logics is to embed a word automaton $\mathcal{W}$, taking care of the linear constraints on the paths, within a tree automaton that, at each step, dispatches copies of $\mathcal{W}$, updated according to the symbol just read, along all the possible branching directions [58,59]. This approach works pretty nicely for deterministic word automata, but not for general nondeterministic ones. The reason is that in order to determine the correct nondeterministic choice at a given time instant one may need to know what choices will be made at future time instants. However, the correctness of the approach can still be recovered if the requirement on determinism is relaxed slightly, allowing for a "controlled" form of nondeterminism. This leads to the notion of nondeterministic *good-for-games automata* [60].

To formally define this concept, we first need to introduce some notation. For an *a priori* fixed set of node degrees $\Lambda \subseteq \mathbb{N}_+$, the *word-on-tree function* wot_Λ: NWA $\to$ NTA maps an NWA $\mathcal{W} = \langle \Sigma, Q, \delta, q_I, Q_F \rangle$ to the NTA $\mathsf{wot}_\Lambda(\mathcal{W}) \triangleq \langle \Sigma, \Lambda, Q, \widehat{\delta}, q_I, Q_F \rangle$, whose tree transition function $\widehat{\delta}$ is derived from the word transition function δ as follows: $\widehat{\delta}(q, \sigma, d) \triangleq \prod_{i=1}^{d} \delta(q, \sigma)$, for all states $q \in Q$, input symbols $\sigma \in \Sigma$, and node degree $d \in \Lambda$. We recall that a Σ-*labelled* Λ-*tree* is a tree, with node degrees (*i.e.*, number of directions/successors of each node) in Λ, equipped with a labelling function from nodes to Σ. Moreover, Trc($\mathcal{T}$) denotes the set of all Σ-traces of $\mathcal{T}$, namely the sequences of labels of paths in the tree.

Definition 9 *(Good-for-games automaton).* Let $\mathcal{T}$ be a class of Σ-labelled Λ-trees. An NWA $\mathcal{W} = \langle \Sigma, Q, \delta, q_I, Q_F \rangle$ is a *good-for-games automaton w.r.t.* $\mathcal{T}$ ($\mathcal{T}$-GFG, for short) if Trc($\mathcal{T}$) $\subseteq$ L($\mathcal{W}$) implies $\mathcal{T} \in$ L($\mathsf{wot}_\Lambda(\mathcal{W})$), for all trees $\mathcal{T} \in \mathcal{T}$.

Intuitively, good-for-games automata only use knowledge of the (non-strict) past to determine the next steps among the available nondeterministic choices and no information on the future. This relates to the notion of strategy in the context of games, where the choice of the actions is purely based on histories (*i.e.*, finite traces). In particular, note that the converse direction, $\mathcal{T} \in$ L($\mathsf{wot}_\Lambda(\mathcal{W})$) implies Trc($\mathcal{T}$) $\subseteq$ L($\mathcal{W}$) always holds true.

In the following we introduce a class of word automata that are GFG for k-bounded trees and will allow us to define a satisfiability algorithm for $\mathrm{SL}^{\mathcal{O}}$ [1G].

5.3. Prefix-deterministic word automata

Prefix-deterministic word automata are NWAs which behave deterministically on arbitrary prefixes of their runs and then act freely, *i.e.*, nondeterministically, afterwards. The aim is to use such automata to encode the checks for compliance of all

the paths of a tree *w.r.t.* an LTL property in a very controlled way. Indeed, we shall take advantage of the prefix-determinism to constrain the nondeterministic choices within the tree automaton to occur only after an initial prefix of a path when all the forks allowed by the input k-fork tree have already occurred. Recall from the definition in Section 2 that an NWA exploits two implicit distinguished rejecting and accepting states $\bot$ and $\top$, respectively.

Definition 10 *(Prefix-deterministic automaton).* An NWA $\mathcal{W} = \langle \Sigma, \mathrm{Q}, \delta, q_I, \mathrm{Q_F} \rangle$ is *prefix-deterministic* (PD-NWA, for short) if there is a deterministic transition function $\widetilde{\delta} \colon \mathrm{Q} \times \Sigma \to \mathrm{Q} \cup \{\bot, \top\}$ with $\widetilde{\delta}(q, \sigma) \in \delta(q, \sigma) \cup \{\bot, \top\}$, for all states $q \in \mathrm{Q}$ and input symbols $\sigma \in \Sigma$, such that the following two properties are equivalent, for all finite words $v \in \Sigma^*$ and infinite words $w \in \Sigma^\omega$:

1) $v \cdot w \in \mathsf{L}(\mathcal{W})$;
2) one of the following holds:

   (i) $q_v = \top$;
   (ii) $q_v \neq \bot$ and $w \in \mathsf{L}(\mathcal{W}_v)$, with $\mathcal{W}_v \triangleq \langle \Sigma, \mathrm{Q}, \delta, q_v, \mathrm{Q_F} \rangle$,

   where $q_v \triangleq \widetilde{\delta}^*(q_I, v)$ is the unique state reached following $\widetilde{\delta}$ from q_I by reading the finite word v.

For every NWA $\mathcal{W}$, we can easily construct a language-equivalent PD-NWA, by using a standard subset construction for the determinisation of the initial behaviours of $\mathcal{W}$ and, then, suitably concatenating it to the original automaton $\mathcal{W}$ to complete the behaviours on the infinite runs.

Theorem 9. *For every* NWA $\mathcal{W}$ *with n states, there exists a* PD-NWA $\mathcal{D}$ *with* $n + 2^n$ *states such that* $\mathsf{L}(\mathcal{D}) = \mathsf{L}(\mathcal{W})$.

Proof. Given an NWA $\mathcal{W} \triangleq \langle \Sigma, \mathrm{Q}, \delta, q_I, \mathrm{Q_F} \rangle$, let us consider the NWA $\mathcal{D} \triangleq \langle \Sigma, \mathrm{Q} \cup 2^{\mathrm{Q}}, \widehat{\delta}, \{q_I\}, \mathrm{Q_F} \rangle$ obtained from $\mathcal{W}$ by extending it with its subset construction as follows:

$$\widehat{\delta}(q, \sigma) \triangleq \delta(q, \sigma), \text{ for all } q \in \mathrm{Q};$$
$$\widehat{\delta}(\mathrm{S}, \sigma) \triangleq \mathrm{S}' \cup \{\mathrm{S}'\}, \text{ where } \mathrm{S}' \triangleq \textstyle\bigcup_{q \in \mathrm{S}} \delta(q, \sigma), \text{ for all } \mathrm{S} \subseteq \mathrm{Q}.$$

Obviously, $\mathcal{D}$ accepts exactly the same infinite words accepted by $\mathcal{W}$. Indeed, every accepting run $q_I, q_1, \ldots$ of $\mathcal{W}$ can be turned immediately into the accepting run $\{q_I\}, q_1, \ldots$ of $\mathcal{D}$. *Vice versa*, notice first that every accepting run of $\mathcal{D}$ has necessarily the form $\{q_I\}, \mathrm{S}_1, \ldots, \mathrm{S}_k, q_{k+1}, q_{k+2}, \ldots$, with $\mathrm{S}_i \subseteq \mathrm{Q}$ and $q_i \in \mathrm{Q}$, for some $k \in \mathbb{N}$, since only the states in $\mathrm{Q_F}$ are accepting. Now, due to the definition of the transition function $\widehat{\delta}$, there must exist a sequence of states $q_I, q_1 \in \mathrm{S}_1, \ldots, q_k \in \mathrm{S}_k$ such that $q_I, q_1, \ldots, q_k, q_{k+1}, q_{k+2}, \ldots$ is a run of $\mathcal{W}$, which is necessarily accepting, since it shares with the original run the same set of states occurring infinitely often.

It only remains to show that $\mathcal{D}$ is a PD-NWA. We take the following function $\widetilde{\delta} \colon (\mathrm{Q} \cup 2^{\mathrm{Q}}) \times \Sigma \to (\mathrm{Q} \cup 2^{\mathrm{Q}}) \cup \{\bot, \top\}$:

$$\widetilde{\delta}(q, \sigma) \triangleq \bot, \text{ for all } q \in \mathrm{Q};$$
$$\widetilde{\delta}(\mathrm{S}, \sigma) \triangleq \mathrm{S}', \text{ where } \mathrm{S}' \triangleq \textstyle\bigcup_{q \in \mathrm{S}} \delta(q, \sigma), \text{ for all } \mathrm{S} \subseteq \mathrm{Q},$$

as the required deterministic transition function. Consider an arbitrary infinite word $v \cdot w \in \mathsf{L}(\mathcal{D})$ and let $\{q_I\}, \mathrm{S}_1, \ldots, \mathrm{S}_k, q_{k+1}, q_{k+2}, \ldots$ be one of its accepting runs. Thanks to the way the transition function $\widehat{\delta}$ is defined, we can always choose *w.l.o.g.* the run in such a way that $k = |v|$. It is easy to see that the sequence $\{q_I\}, \mathrm{S}_1, \ldots, \mathrm{S}_k$ complies with the deterministic transition function $\widetilde{\delta}$, *i.e.*, $\mathrm{S}_k \triangleq \widetilde{\delta}^*(\{q_I\}, v) \neq \bot$. Moreover, $\mathrm{S}_k, q_{k+1}, q_{k+2}, \ldots$ is an accepting run of $\mathcal{D}_v \triangleq \langle \Sigma, \mathrm{Q} \cup 2^{\mathrm{Q}}, \widehat{\delta}, \mathrm{S}_k, \mathrm{Q_F} \rangle$ on w. Hence, $w \in \mathsf{L}(\mathcal{D}_v)$, as required by definition of PD-NWA.

Vice versa, let $v \in \Sigma^*$ and $w \in \Sigma^\omega$ be two words such that $q_v \triangleq \widetilde{\delta}^*(\{q_I\}, v) \neq \bot$ and $w \in \mathsf{L}(\mathcal{D}_v)$ with $\mathcal{D}_v \triangleq \langle \Sigma, \mathrm{Q} \cup 2^{\mathrm{Q}}, \widehat{\delta}, q_v, \mathrm{Q_F} \rangle$. Obviously, there exist a finite run $\{q_I\}, \mathrm{S}_1, \ldots, \mathrm{S}_k$ of $\mathcal{D}$ on v, with $\mathrm{S}_k = q_v$ and $k = |v|$, and an infinite accepting run $q_v, q_{k+1}, q_{k+2}, \ldots$ of $\mathcal{D}_v$ on w. Hence, $\{q_I\}, \mathrm{S}_1, \ldots, \mathrm{S}_k, q_{k+1}, q_{k+2}, \ldots$ is an infinite accepting run of $\mathcal{D}$ on $v \cdot w$, which implies $v \cdot w \in \mathsf{L}(\mathcal{D})$, again as required by definition of PD-NWA. □

The standard automata construction for LTL in [61] can easily be lifted to PD-NWA, as stated by Theorem 10. Note that, when the WLTL fragment of LTL is considered, the automaton construction needs to deal with a number of states that is only singly-exponential *w.r.t.* the size of the input formula.

Theorem 10. *For every* LTL *(*resp., WLTL*) formula* ψ*, there is a* PD-NWA $\mathcal{D}_\psi$ *with* $2^{\mathrm{O}(2^{|\psi|})}$ *(*resp., $\mathrm{O}\left(2^{|\psi|^3}\right)$*) states such that* $\mathsf{L}(\mathcal{D}_\psi) = \mathsf{L}(\psi)$.

Proof. The proof for LTL formulae immediately follows by applying Theorem 9 to the NWA obtained by means of the Vardi-Wolper construction in [61]. As far as WLTL is concerned, we construct, instead, an *ad hoc* PD-NWA that extends the Vardi-Wolper automaton with a deterministic transition function unravelling the one-step unfolding of the input formula.

Before proceeding, we need to introduce some notation. With $\mathring{\wedge}$ and $\mathring{\vee}$ we denote the binary Boolean functions that simplify the arguments of the corresponding Boolean connectives whenever possible. Formally:

$$\psi_1 \mathring{\vee} \psi_2 \triangleq \begin{cases} \top, & \text{if } \psi_1 = \top \text{ or } \psi_2 = \top; \\ \psi_1, & \text{if } \psi_2 = \bot; \\ \psi_2, & \text{if } \psi_1 = \bot; \\ \psi_1 \vee \psi_2, & \text{otherwise;} \end{cases} \qquad \psi_1 \mathring{\wedge} \psi_2 \triangleq \begin{cases} \bot, & \text{if } \psi_1 = \bot \text{ or } \psi_2 = \bot; \\ \psi_1, & \text{if } \psi_2 = \top; \\ \psi_2, & \text{if } \psi_1 = \top; \\ \psi_1 \wedge \psi_2, & \text{otherwise.} \end{cases}$$

By denoting with LTL^+ the set of LTL formulae in positive normal form, the function $\gamma : \mathrm{LTL}^+ \times 2^{\mathrm{AP}} \to \mathrm{LTL}^+$ defined below computes, by means of the standard one-step unfolding of the temporal operators $\mathtt{U}$ and $\mathtt{R}$, the *one-step evaluation* of the LTL^+ formula given in input *w.r.t.* some set $\sigma \in 2^{\mathrm{AP}}$ of atomic propositions. The idea here is that σ encodes a possible assignment for the atomic propositions and γ partially evaluates the temporal formula with respect to that local assignment.

- $\gamma(\top, \sigma) \triangleq \top$ and $\gamma(\bot, \sigma) \triangleq \bot$;
- $\gamma(p, \sigma) \triangleq \top$, $\gamma(\neg p, \sigma) \triangleq \bot$, if $p \in \sigma$;
- $\gamma(p, \sigma) \triangleq \bot$, $\gamma(\neg p, \sigma) \triangleq \top$, if $p \notin \sigma$;
- $\gamma(\psi_1 \wedge \psi_2, \sigma) \triangleq \gamma(\psi_1, \sigma) \mathring{\wedge} \gamma(\psi_2, \sigma)$;
- $\gamma(\psi_1 \vee \psi_2, \sigma) \triangleq \gamma(\psi_1, \sigma) \mathring{\vee} \gamma(\psi_2, \sigma)$;
- $\gamma(\mathtt{X}\,\psi, \sigma) \triangleq \psi$;
- $\gamma(\psi_1 \,\mathtt{U}\, \psi_2, \sigma) \triangleq \gamma(\psi_2 \vee (\psi_1 \wedge \mathtt{X}(\psi_1 \,\mathtt{U}\, \psi_2)), \sigma)$;
- $\gamma(\psi_1 \,\mathtt{R}\, \psi_2, \sigma) \triangleq \gamma(\psi_2 \wedge (\psi_1 \vee \mathtt{X}(\psi_1 \,\mathtt{R}\, \psi_2)), \sigma)$.

The one-step evaluation can be lifted to arbitrary finite sequences of sets of atomic propositions (*i.e.*, finite-length temporal assignments) in the obvious way to obtain the *partial evaluation function* $\gamma^* : \mathrm{LTL}^+ \times (2^{\mathrm{AP}})^* \to \mathrm{LTL}^+$ defined as follows:

- $\gamma^*(\psi, \varepsilon) \triangleq \psi$;
- $\gamma^*(\psi, \varsigma \cdot \vec{\sigma}) \triangleq \gamma^*(\gamma(\psi, \varsigma), \vec{\sigma})$.

The $\mathtt{X}$-depth of an LTL formula is given by the function $\mathsf{dep}_{\mathtt{X}} : \mathrm{LTL}^+ \to \mathbb{N}$:

- $\mathsf{dep}_{\mathtt{X}}(\bot) = \mathsf{dep}_{\mathtt{X}}(\top) = \mathsf{dep}_{\mathtt{X}}(p) = \mathsf{dep}_{\mathtt{X}}(\neg p) \triangleq 0$;
- $\mathsf{dep}_{\mathtt{X}}(\psi_1 \odot \psi_2) \triangleq \max\{\mathsf{dep}_{\mathtt{X}}(\psi_1), \mathsf{dep}_{\mathtt{X}}(\psi_2)\}$, where $\odot \in \{\wedge, \vee, \mathtt{U}, \mathtt{R}\}$;
- $\mathsf{dep}_{\mathtt{X}}(\mathtt{X}\,\psi) \triangleq 1 + \mathsf{dep}_{\mathtt{X}}(\psi)$.

We shall denote with $\Sigma^{\leq k}$ (*resp.*, $\Sigma^{>k}$), for an integer $k \in \mathbb{N}$, the set of finite words over Σ of length less than or equal to (*resp.*, greater than) k.

The function $\mathsf{isub} : \mathrm{LTL}^+ \to 2^{\mathrm{LTL}^+}$ allows us to determine for a LTL formula ψ given in input the *induced subformulae* of ψ that can be obtained by partial evaluation *w.r.t.* all the possible temporal assignments of length bounded by the $\mathtt{X}$-depth of the formula ψ:

- $\mathsf{isub}(\bot) \triangleq \{\bot\}$ and $\mathsf{isub}(\top) \triangleq \{\top\}$;
- $\mathsf{isub}(p) \triangleq \{\bot, \top, p\}$ and $\mathsf{isub}(\neg p) \triangleq \{\bot, \top, \neg p\}$;
- $\mathsf{isub}(\psi_1 \odot \psi_2) \triangleq \{\psi_1 \odot \psi_2\} \cup \left\{\theta_1 \mathring{\odot} \theta_2 \,\middle|\, \theta_1 \in \mathsf{isub}(\psi_1) \wedge \theta_2 \in \mathsf{isub}(\psi_2)\right\}$, with $\odot \in \{\wedge, \vee\}$;
- $\mathsf{isub}(\mathtt{X}\,\psi) \triangleq \{\mathtt{X}\,\psi\} \cup \mathsf{isub}(\psi)$;
- $\mathsf{isub}(\psi_1 \odot \psi_2) \triangleq \left\{\gamma^*(\psi_1 \odot \psi_2, \vec{\sigma}) \,\middle|\, \vec{\sigma} \in (2^{\mathsf{ap}(\psi_1 \odot \psi_2)})^{\leq 1 + \mathsf{dep}_{\mathtt{X}}(\psi_1 \odot \psi_2)}\right\}$, with $\odot \in \{\mathtt{U}, \mathtt{R}\}$.

At this point, given a WLTL formula ψ, let $\Gamma_\psi \triangleq \left\{\gamma^*(\psi, \vec{\sigma}) \,\middle|\, \vec{\sigma} \in (2^{\mathsf{ap}(\psi)})^*\right\}$ be the set of all its possible partial evaluations of ψ. Then, the following claim can be shown by analysing the definition of the partial evaluation function and the set of induced subformulae.

Claim 2. *For all* WLTL *formulae* ψ_1, ψ_2, *and* ψ, *it holds that:*

(a) $\gamma^*(\psi, \vec{\sigma}) \in \{\bot, \top\}$, *for all* $\vec{\sigma} \in (2^{\mathsf{ap}(\psi)})^+$, *whenever* $\psi \in \{\bot, \top\} \cup \mathsf{Lit}$;
(b) $\gamma^*(\psi, \vec{\sigma}) = \gamma^*(\psi_1, \vec{\sigma}) \mathring{\odot} \gamma^*(\psi_2, \vec{\sigma})$, *for all* $\vec{\sigma} \in (2^{\mathsf{ap}(\psi)})^+$, *whenever* $\psi = \psi_1 \odot \psi_2$ *with* $\odot \in \{\wedge, \vee\}$;
(c) $\gamma^*(\psi_1 \,\mathtt{U}\, \psi_2, \vec{\sigma}) = \gamma^*(\psi_2, \vec{\sigma}) \mathring{\vee} (\gamma^*(\psi_1, \vec{\sigma}) \mathring{\wedge} \gamma^*(\mathtt{X}(\psi_1 \,\mathtt{U}\, \psi_2), \vec{\sigma}))$, *for all* $\vec{\sigma} \in (2^{\mathsf{ap}(\psi)})^+$;
(d) $\gamma^*(\psi_1 \,\mathtt{R}\, \psi_2, \vec{\sigma}) = \gamma^*(\psi_2, \vec{\sigma}) \mathring{\wedge} (\gamma^*(\psi_1, \vec{\sigma}) \mathring{\vee} \gamma^*(\mathtt{X}(\psi_1 \,\mathtt{R}\, \psi_2), \vec{\sigma}))$, *for all* $\vec{\sigma} \in (2^{\mathsf{ap}(\psi)})^+$;
(e) $\gamma^*(\psi, \vec{\sigma}) \in \{\bot, \top\}$, *for all* $\vec{\sigma} \in (2^{\mathsf{ap}(\psi)})^{> \mathsf{dep}_{\mathtt{X}}(\psi)}$, *whenever* ψ *is* $\mathtt{U}/\mathtt{R}$*-free;*
(f) $\gamma^*(\psi, \vec{\sigma} \cdot \vec{\sigma}') \in \{\bot, \top, \gamma^*(\psi, \vec{\sigma}')\}$, *for all* $\vec{\sigma} \in (2^{\mathsf{ap}(\psi)})^+$ *and* $\vec{\sigma}' \in (2^{\mathsf{ap}(\psi)})^{\mathsf{dep}_{\mathtt{X}}(\psi)}$, *whenever* $\psi = \psi_1 \odot \psi_2$ *with* $\odot \in \{\mathtt{U}, \mathtt{R}\}$, *where both* ψ_1 *and* ψ_2 *are* $\mathtt{U}/\mathtt{R}$*-free;*
(g) $\gamma^*(\psi, \vec{\sigma}) \in \mathsf{isub}(\psi)$, *for all* $\vec{\sigma} \in (2^{\mathsf{ap}(\psi)})^*$, *i.e.*, $\Gamma_\psi \subseteq \mathsf{isub}(\psi)$;
(h) $|\mathsf{isub}(\psi)| = \mathrm{O}\left(2^{|\psi|^3}\right)$.

Proof. Each property in the statement can be proved by means of an inductive proof on the length of the finite word $\vec{\sigma} \in (2^{\mathsf{ap}(\psi)})^+$, on the structure of the WLTL formula ψ, or on a suitable combination thereof.

- **[Property (a)]** The statement can be derived by a trivial induction on $\vec{\sigma}$.
- **[Property (b)]** The base case $\sigma \in 2^{\mathsf{ap}(\psi)}$ is an immediate consequence of the definition of the one-step evaluation and partial evaluation functions, since $\gamma^*(\psi, \sigma) = \gamma^*(\gamma(\psi, \sigma), \varepsilon) = \gamma(\psi, \sigma)$. The inductive case is derived from the following equalities, where the inductive hypothesis is applied in the third derivation step, with $\vec{\sigma} = \varsigma \cdot \vec{\sigma}'$:

$$\begin{aligned}
\gamma^*(\psi_1 \odot \psi_2, \varsigma \cdot \vec{\sigma}') &= \gamma^*(\gamma(\psi_1 \odot \psi_2, \varsigma), \vec{\sigma}') \\
&= \gamma^*(\gamma(\psi_1, \varsigma) \mathbin{\dot{\odot}} \gamma(\psi_2, \varsigma), \vec{\sigma}') \\
&= \gamma^*(\gamma(\psi_1, \varsigma), \vec{\sigma}') \mathbin{\dot{\odot}} \gamma^*(\gamma(\psi_2, \varsigma), \vec{\sigma}') \\
&= \gamma^*(\psi_1, \varsigma \cdot \vec{\sigma}') \mathbin{\dot{\odot}} \gamma^*(\psi_2, \varsigma \cdot \vec{\sigma}').
\end{aligned}$$

- **[Properties (c) and (d)]** The statements can be derived by a trivial induction on the length of $\vec{\sigma}$, where the inductive case exploits Property (b).
- **[Property (e)]** The base cases $\psi \in \{\bot, \top\} \cup \mathrm{Lit}$ and the inductive cases $\psi = \psi_1 \odot \psi_2$ with $\odot \in \{\wedge, \vee\}$ follow from Properties (a) and (b), respectively. Thus, in the following equalities we only consider the case $\psi = \mathtt{X}\,\psi'$ with $\vec{\sigma} = \varsigma \cdot \vec{\sigma}' \in (2^{\mathsf{ap}(\psi)})^{>\mathsf{dep}_{\mathtt{X}}(\psi)} = (2^{\mathsf{ap}(\psi)})^{>1+\mathsf{dep}_{\mathtt{X}}(\psi')}$, where the inductive hypothesis is used in the third derivation step:

$$\begin{aligned}
\gamma^*(\mathtt{X}\,\psi', \varsigma \cdot \vec{\sigma}') &= \gamma^*(\gamma(\mathtt{X}\,\psi', \varsigma), \vec{\sigma}') \\
&= \gamma^*(\psi', \vec{\sigma}') \\
&\in \{\bot, \top\}.
\end{aligned}$$

- **[Property (f)]** We show the property only for the until operator, the case of the release operator being virtually identical. By Property (c), we have that $\gamma^*(\psi_1 \,\mathtt{U}\, \psi_2, \vec{\sigma} \cdot \vec{\sigma}') = \gamma^*(\psi_2, \vec{\sigma} \cdot \vec{\sigma}') \mathbin{\dot{\vee}} (\gamma^*(\psi_1, \vec{\sigma} \cdot \vec{\sigma}') \mathbin{\dot{\wedge}} \gamma^*(\mathtt{X}(\psi_1 \,\mathtt{U}\, \psi_2), \vec{\sigma} \cdot \vec{\sigma}'))$. Moreover, by Property (e), it holds that $\gamma^*(\psi_1, \vec{\sigma} \cdot \vec{\sigma}'), \gamma^*(\psi_2, \vec{\sigma} \cdot \vec{\sigma}') \in \{\bot, \top\}$, since ψ_1 and ψ_2 are $\mathtt{U}/\mathtt{R}$-free and $\vec{\sigma} \cdot \vec{\sigma}' \in (2^{\mathsf{ap}(\psi)})^{>\max\{\mathsf{dep}_{\mathtt{X}}(\psi_1), \mathsf{dep}_{\mathtt{X}}(\psi_2)\}} = (2^{\mathsf{ap}(\psi)})^{>\mathsf{dep}_{\mathtt{X}}(\psi_1 \,\mathtt{U}\, \psi_2)}$. Hence, $\gamma^*(\psi_1 \,\mathtt{U}\, \psi_2, \vec{\sigma} \cdot \vec{\sigma}') \in \{\bot, \top\}$, if $\gamma^*(\psi_2, \vec{\sigma} \cdot \vec{\sigma}') = \top$ or $\gamma^*(\psi_1, \vec{\sigma} \cdot \vec{\sigma}') = \bot$, and $\gamma^*(\psi_1 \,\mathtt{U}\, \psi_2, \vec{\sigma} \cdot \vec{\sigma}') = \gamma^*(\mathtt{X}(\psi_1 \,\mathtt{U}\, \psi_2), \vec{\sigma} \cdot \vec{\sigma}')$, otherwise. In the first case we have reached the desired conclusion. As to the second case, the thesis can be proved by induction on the length of $\vec{\sigma}$, where the inductive case $\vec{\sigma} = \varsigma \cdot \vec{\sigma}''$ proceeds as follows, where the inductive hypothesis is used in the last derivation step:

$$\begin{aligned}
\gamma^*(\psi_1 \,\mathtt{U}\, \psi_2, \varsigma \cdot \vec{\sigma}'' \cdot \vec{\sigma}') &= \gamma^*(\mathtt{X}(\psi_1 \,\mathtt{U}\, \psi_2), \varsigma \cdot \vec{\sigma}'' \cdot \vec{\sigma}') \\
&= \gamma^*(\gamma(\mathtt{X}(\psi_1 \,\mathtt{U}\, \psi_2), \varsigma), \vec{\sigma}'' \cdot \vec{\sigma}') \\
&= \gamma^*(\psi_1 \,\mathtt{U}\, \psi_2, \vec{\sigma}'' \cdot \vec{\sigma}') \\
&= \gamma^*(\psi_1 \,\mathtt{U}\, \psi_2, \vec{\sigma}').
\end{aligned}$$

- **[Property (g)]** The property can be derived by structural induction on ψ, exploiting Properties (a)-(f).
- **[Property (h)]** The last property can also be proved by structural induction on ψ, where we consider the $\mathtt{U}/\mathtt{R}$-formulae as base cases. Since the base cases $\psi \in \{\bot, \top\} \cup \mathrm{Lit}$ and the inductive case $\psi = \mathtt{X}\,\psi'$ are simple, we only prove the base cases $\psi = \psi_1 \odot \psi_2$, with $\odot \in \{\mathtt{U}, \mathtt{R}\}$, and the inductive cases $\psi = \psi_1 \odot \psi_2$, with $\odot \in \{\wedge, \vee\}$:

  - **[$\odot \in \{\mathtt{U}, \mathtt{R}\}$]**

$$\begin{aligned}
|\mathsf{isub}(\psi_1 \odot \psi_2)| &= \left| \left\{ \gamma^*(\psi_1 \odot \psi_2, \vec{\sigma}) \;\middle|\; \vec{\sigma} \in (2^{\mathsf{ap}(\psi_1 \odot \psi_2)})^{\leq 1+\mathsf{dep}_{\mathtt{X}}(\psi_1 \odot \psi_2)} \right\} \right| \\
&\leq \left| (2^{\mathsf{ap}(\psi_1 \odot \psi_2)})^{\leq 1+\mathsf{dep}_{\mathtt{X}}(\psi_1 \odot \psi_2)} \right| \\
&= 2^{|\mathsf{ap}(\psi_1 \odot \psi_2)| \cdot \frac{1}{2} \cdot (1+\mathsf{dep}_{\mathtt{X}}(\psi_1 \odot \psi_2))(2+\mathsf{dep}_{\mathtt{X}}(\psi_1 \odot \psi_2))} \\
&\leq 2^{|\psi_1 \odot \psi_2| \cdot \frac{1}{2} \cdot (1+|\psi_1 \odot \psi_2|)(2+|\psi_1 \odot \psi_2|)} \\
&= O\left(2^{|\psi_1 \odot \psi_2|^3}\right).
\end{aligned}$$

- **[$\odot \in \{\wedge, \vee\}$]**

$$\begin{aligned}
|\mathsf{isub}(\psi_1 \odot \psi_2)| &= \left|\{\psi_1 \odot \psi_2\} \cup \left\{\theta_1 \mathring{\odot} \theta_2 \,\middle|\, \theta_1 \in \mathsf{isub}(\psi_1) \wedge \theta_2 \in \mathsf{isub}(\psi_2)\right\}\right| \\
&\leq 1 + \left|\left\{\theta_1 \mathring{\odot} \theta_2 \,\middle|\, \theta_1 \in \mathsf{isub}(\psi_1) \wedge \theta_2 \in \mathsf{isub}(\psi_2)\right\}\right| \\
&\leq 1 + |\mathsf{isub}(\psi_1)| \cdot |\mathsf{isub}(\psi_2)| \\
&= \mathrm{O}\Big(1 + \mathrm{O}\Big(2^{|\psi_1|^3}\Big) + \mathrm{O}\Big(2^{|\psi_2|^3}\Big)\Big) \\
&= \mathrm{O}\Big(2^{|\psi_1 \odot \psi_2|^3}\Big). \quad \square
\end{aligned}$$

To conclude, let $\mathcal{W}_\psi \triangleq \langle 2^{\mathrm{AP}}, \mathrm{Q}, \delta, q_I, \mathrm{Q_F}\rangle$ be the Vardi-Wolper automaton of the WLTL formula ψ and consider the NWA $\mathcal{D}_\psi \triangleq \langle 2^{\mathrm{AP}}, \mathrm{Q} \cup \Gamma_\psi, \widehat{\delta}, \psi, \mathrm{Q_F}\rangle$ whose transition function $\widehat{\delta}$ is defined as follows, where by $q \Vdash \gamma(\theta, \sigma)$ we mean that the set of subformulae q of the Vardi-Wolper construction satisfies the LTL formula $\gamma(\theta, \sigma)$, once the temporal operators are interpreted as fresh atomic propositions:

$$\begin{aligned}
&\widehat{\delta}(q, \sigma) \triangleq \delta(q, \sigma), \text{ for all } q \in \mathrm{Q}; \\
&\widehat{\delta}(\theta, \sigma) \triangleq \left\{q \in \mathrm{Q} \,\middle|\, q \Vdash \gamma(\theta, \sigma)\right\} \cup \{\gamma(\theta, \sigma)\}, \text{ for all } \theta \in \Gamma_\psi.
\end{aligned}$$

Intuitively, the partial evaluation replaces the subset construction of Theorem 9 and ensures that the automaton can behave deterministically, by choosing at each such step the unique successor $\gamma(\theta, \sigma)$, from state θ on reading the symbol σ. Essentially, γ encodes the deterministic transition function of the PD-NWA. Clearly, in order to accept the infinite input the automaton will need to move to its non-deterministic component at some point, since γ moves from Γ_ψ to Γ_ψ and this set has empty intersection with $\mathrm{Q_F}$. Note that, thanks to Items g and h of the previous claim, $\mathcal{D}_\psi$ has the required size. To formally prove that $\mathcal{D}_\psi$ is a PD-NWA with $\mathsf{L}(\mathcal{D}_\psi) = \mathsf{L}(\mathcal{D}_\psi) = \mathsf{L}(\psi)$ one can apply, *mutatis mutandis*, the same approach used in the proof of Theorem 9. $\square$

Observe that the approach we used to construct a PD-NWA for WLTL could, in principle, be used for full LTL. However, there is no advantage in doing that compared to the simpler application of Theorem 9 to the Vardi-Wolper construction, since the asymptotic bound would remain unvaried. The following theorem shows, indeed, that for LTL one cannot hope for smaller PD-NWA, since there are LTL formulae whose smallest PD-NWA are doubly-exponential in their size.

Theorem 11. *For every $n \in \mathbb{N}$, there is an* LTL *formula ψ_n, with $|\psi_n| = \Theta(n^2)$, whose smallest* PD-NWA *$\mathcal{D}_n$ with $\mathsf{L}(\mathcal{D}_n) = \mathsf{L}(\psi_n)$ (*resp.*, $\mathsf{L}(\mathcal{D}_n) = \mathsf{L}(\neg\psi_n)$) has at least 2^{2^n} states.*

Proof. We prove the result by first identifying a class $\{\mathrm{L}_n\}_{n\in\mathbb{N}}$ of ω-languages that can be recognised by PD-NWA with at least a double-exponential number of states and then showing that the class can be characterised by an LTL formula of size quadratic in n.

The family we consider here is similar to the one used to show that deterministic (Müller) automata are doubly-exponentially less succinct than alternating (weak) automata [62]. Let $\Sigma \triangleq \{0, 1, \|, \#\}$ be the alphabet. For every index $n \in \mathbb{N}$, consider the ω-language $\mathrm{L}_n \subseteq \Sigma^\omega$ containing all and only the infinite words $w \in \Sigma^\omega$ satisfying the following property: there exist two words $x \in \Sigma^*$ and $y \in \Sigma^\omega$ such that $w = x \,\|\, z\, v \,\#\, z\, y$, for some pair of additional finite words $z \in \{0, 1\}^n$ and $v \in \{0, 1, \|\}^*$. Obviously, there are 2^{2^n} sets of words $\mathrm{Z} \subseteq \{0, 1\}^n$. For each such set Z of cardinality $k \triangleq |\mathrm{Z}|$, there exists a finite word w_Z of the form $\| z_1 \| \cdots \| z_k$, where $z_1 \cdots z_k$ is an arbitrary permutation of its words. Note that $w_\emptyset$ is the one-symbol word $\|$. The following claim is proven to hold for the class $\{\mathrm{L}_n\}_{n\in\mathbb{N}}$.

Claim 3. *For every $n \in \mathbb{N}$, the smallest* PD-NWA *$\mathcal{D}_n$ such that $\mathsf{L}(\mathcal{D}_n) = \mathrm{L}_n$ (*resp.*, $\mathsf{L}(\mathcal{D}_n) = \overline{\mathrm{L}_n}$) has at least 2^{2^n} states.*

Proof. Suppose, by contradiction, that L_n (*resp.*, $\overline{\mathrm{L}_n}$) is recognised by a PD-NWA $\mathcal{D}$ with strictly less than 2^{2^n} states and let $\widetilde{\delta}\colon \mathrm{Q} \times \Sigma \to \mathrm{Q} \cup \{\bot, \top\}$ be any one of deterministic transition functions witnessing the fact that $\mathcal{D}$ is PD-NWA. By the pigeonhole principle, there exist two different sets $\mathrm{Z}_1, \mathrm{Z}_2 \subseteq \{0, 1\}^n$ such that $\widetilde{\delta}^*(q_I, w_{\mathrm{Z}_1}) = \widetilde{\delta}^*(q_I, w_{\mathrm{Z}_2})$. *W.l.o.g.*, assume $\mathrm{Z}_1 \setminus \mathrm{Z}_2 \neq \emptyset$ and let $z \in \mathrm{Z}_1 \setminus \mathrm{Z}_2$. Obviously, $w_{\mathrm{Z}_1} = x \,\|\, zv$, for some $x \in \Sigma^\omega$ and $v \in \{0, 1, \|\}^*$. Hence, $w_1 \triangleq w_{\mathrm{Z}_1} \,\#\, z \,\#^\omega \in \mathrm{L}_n$, since $w_{\mathrm{Z}_1} \,\#\, z \,\#^\omega = x \,\|\, z\, v \,\#\, z\, y$, with $y = \#^\omega$. On the contrary, $w_2 \triangleq w_{\mathrm{Z}_2} \,\#\, z \,\#^\omega \notin \mathrm{L}_n$, since w_{Z_2} does not contain z. However, since the automaton ends up in the same state after reading w_{Z_1} and w_{Z_2}, and w_1 and w_2 do not differ after those two prefixes, $\mathcal{D}$ either accepts both words or none of them, which contradicts the assumption $\mathsf{L}(\mathcal{D}_n) = \mathrm{L}_n$ (*resp.*, $\mathsf{L}(\mathcal{D}_n) = \overline{\mathrm{L}_n}$). $\square$

To complete the proof of the theorem it suffices to observe that each language of the family $\{\mathrm{L}_n\}_{n\in\mathbb{N}}$ precisely corresponds to the language $\mathsf{L}(\psi_n)$ recognised by the LTL formula

$$\psi_n \triangleq \mathrm{F}\left(\| \wedge \bigwedge_{i=1}^{n} \mathrm{X}^i\left(\bigvee_{j\in\{0,1\}}\left(j \wedge (\neg\#)\,\mathrm{U}\,(\# \wedge \mathrm{X}^i j)\right)\right)\right)$$

built taking Σ as the set of atomic propositions. The formula essentially checks that the infinite word contains at some point two identical copies of the same n-bit sequence separated by a #-less sequence of symbols followed by a single #. This is precisely the form of the words in $\mathsf{L}(\psi_n)$.

It is easy to see that the formula has size $|\psi_n| = \Theta(n^2)$, due to having n conjuncts with increasing number of nested next operators. Hence, the stated results follow as an immediate consequence of the claim. □

The following result ensures that every PD-NWA can be embedded within a bounded-fork tree automaton, a result that will be leveraged in the next section.

Theorem 12. *Every* PD-NWA *is* GFG w.r.t. *the class of bounded-fork trees.*

Proof. Let $\mathcal{W}$ be a PD-NWA and $\mathcal{T}$ a bounded-fork Λ-tree such that $\mathrm{Trc}(\mathcal{T}) \subseteq \mathsf{L}(\mathcal{W})$. We want to show that $\mathcal{T} \in \mathsf{L}(\mathrm{wot}_\Lambda(\mathcal{W}))$. Since $\mathcal{T}$ is bounded-fork, there necessarily exists a finite subtree $\mathcal{T}'$ containing all forks of $\mathcal{T}$. Now, let ϑ' be the finite run of $\mathrm{wot}_\Lambda(\mathcal{W})$ over $\mathcal{T}'$ obtained by using any one of the possible deterministic transition functions $\widetilde{\delta}\colon \mathrm{Q}\times\Sigma\to\mathrm{Q}\cup\{\bot,\top\}$ of $\mathcal{W}$. Since $\mathrm{Trc}(\mathcal{T}) \subseteq \mathsf{L}(\mathcal{W})$, this finite run can be extended to an infinite accepting run ϑ of $\mathrm{wot}_\Lambda(\mathcal{W})$ over $\mathcal{T}$ by using the full nondeterministic transition function δ of $\mathcal{W}$. Obviously, this can always be done since $\mathcal{W}$ is a PD-NWA. □

6. The satisfiability problem

We can finally address the solution of the satisfiability problem for $\mathrm{SL}^{\not\ni}[1\mathrm{G}]$ by reducing it to the non-emptiness problem of a suitable bounded-fork automaton. To proceed, let φ be an arbitrary $\mathrm{SL}^{\not\ni}[1\mathrm{G}]$ sentence and, thanks to Proposition 2, $\eth \triangleq \langle\zeta, \Phi, \ell\rangle$ its corresponding Skolem skeleton. Corollary 2 tells us that φ is satisfiable *iff* $\eth$ is single-time normally satisfied by a k-fork CGS, with $k \triangleq \max\{0, |\Phi| - 1\}$. We thus construct a k-NTA $\mathcal{N}_\varphi$ recognising all normal models of $\eth$, which are, therefore, also models of φ. The automaton is the product $\mathcal{N}_\varphi \triangleq \mathcal{D}_\zeta \times \mathcal{D}_\eth \times \mathcal{N}_\Phi$ of the following three components:

(1) $\mathcal{D}_\zeta$ is a trivial single-state safety automaton checking whether the labelling of the root satisfies the Boolean formula ζ;
(2) $\mathcal{D}_\eth$ is a bounded-fork deterministic safety automaton ensuring the compliance of the structure of the tree in input with Definition 7 of normal model;
(3) the nondeterministic Büchi automaton $\mathcal{N}_\Phi$ verifies that all paths identified by the binding $\flat$ of some sentence $\forall\flat\psi$ in Φ satisfy the LTL formula ψ.

We focus on the definitions of $\mathcal{D}_\eth$ and $\mathcal{N}_\Phi$ only, $\mathcal{D}_\zeta$ being obvious.

Before proceeding, we need to fix some notation. Let $\mathrm{A} \subseteq \mathrm{AP}$ and $\mathrm{B} \subseteq \mathrm{Bn}(\mathcal{F})$ be the sets of all the atomic propositions and binding prefixes occurring in some sentence of the universal skeleton $\eth$, respectively. The automaton alphabet $\Sigma \subseteq 2^{\mathrm{A}\cup\mathrm{B}}$ is then the set of all those symbols $\sigma \subseteq \mathrm{A}\cup\mathrm{B}$ satisfying the following local coherence condition: if $\ell(\phi) \in \sigma$ then $\flat \in \sigma$, for all universal sentences $\phi \triangleq \forall\flat\psi \in \Phi$.

The idea is to recognise all normal models whose labelling is enriched with the bindings of the sentences that are satisfied along some path through each node, as prescribed by Items 2c and 2d of Definition 7. In particular, the local coherence condition precisely corresponds to the property required on the local binding function l by Item 2a of the same definition. Obviously, the branching degree of the tree underlying the normal model is bounded by $|\mathsf{muc}(\mathrm{B})|$, thus, the set of node degrees is $\Lambda \triangleq [1, |\mathsf{muc}(\mathrm{B})|]$. Finally, let us consider an arbitrary function $\widehat{\mathsf{r}}\colon \mathrm{X} \in 2^{\mathrm{B}} \mapsto ([1, |\mathsf{muc}(\mathrm{X})|] \to \mathsf{muc}(\mathrm{X}))$ such that, for each set of bindings $\mathrm{X} \subseteq \mathrm{B}$, the associated map $\widehat{\mathsf{f}} \triangleq \widehat{\mathsf{r}}(\mathrm{X})\colon [1, |\mathsf{muc}(\mathrm{X})|] \to \mathsf{muc}(\mathrm{X})$ is a bijection between indices and set of maximally unifying bindings. This function is used in the following construction to specifically ensure the condition stated in Item 2b of Definition 7.

Construction 1 *(Structure automaton). The* structure automaton $\mathcal{D}_\eth$ *is the safety* DTA $\langle\Sigma, \Lambda, 2^{\mathrm{B}} \times [0,k], \delta, (\mathrm{B}, k)\rangle$*, whose transition function* δ *is defined as follows:*

$$\delta((\mathrm{X},h),\sigma,d) \triangleq \begin{cases} \prod_{i=1}^{d} (\widehat{\mathsf{f}}(i), h-1), & \text{if } \mathrm{X} = \sigma\cap\mathrm{B},\ h>0, \text{ and } d = \left|\mathsf{img}\left(\widehat{\mathsf{f}}\right)\right| > 1;\\ (\mathrm{X},h), & \text{if } \mathrm{X} = \sigma\cap\mathrm{B} \text{ and } d = \left|\mathsf{img}\left(\widehat{\mathsf{f}}\right)\right| = 1;\\ \bot, & \text{otherwise;}\end{cases}$$

where $\widehat{\mathsf{f}} \triangleq \widehat{\mathsf{r}}(\sigma\cap\mathrm{B})$*, for any set of bindings* $\mathrm{X}\subseteq\mathrm{B}$*, index* $h \in [0,k]$*, input symbol* $\sigma\in\Sigma$*, and node degree* $d\in\Lambda$*.*

By construction, thanks to the second component of the states, it is immediate to see that $\mathcal{D}_\eth$ enjoys the bounded-fork property. Let $\mathrm{SL}^{\not\ni}_{\mathrm{K}}[1\mathrm{G}]$ (*resp.*, $\mathrm{WSL}^{\not\ni}_{\mathrm{K}}[1\mathrm{G}]$), for $k \in \mathbb{N}$, denote the fragment of $\mathrm{SL}^{\not\ni}[1\mathrm{G}]$ (*resp.*, $\mathrm{WSL}^{\not\ni}[1\mathrm{G}]$) containing only

sentences with at most $k+1$ distinct occurrences of principal subsentences. Recall that, thanks to Corollary 2, the number k coincides with the bound on the number of forks of the underlying model. Hence, we immediately obtain the following result.

Proposition 3. *The* DTA $\mathcal{D}_{\mathfrak{F}}$ *is* k*-forking, for every* $\mathrm{SL}^{\not\ni}_{k}[1\mathrm{G}]$ *skeleton* $\mathfrak{F}$.

It can be shown that, if we extend any normal model of $\mathfrak{F}$ with the binding labelling dictated by its global binding function, we obtain a tree structure that is accepted by $\mathcal{D}_{\mathfrak{F}}$. *Vice versa*, every tree accepted by $\mathcal{D}_{\mathfrak{F}}$ is the backbone of a tree CGS, whose functions l, g, and r (see Definition 7) can easily be extracted from the labelling. To ensure that this CGS is actually a normal model, we need to verify that the paths labelled by some binding $\flat$ satisfy the corresponding sentences in Φ. This is precisely the goal of $\mathcal{N}_{\Phi}$.

By Theorem 10, for any $\mathrm{SL}^{\not\ni}[1\mathrm{G}]$ (*resp.*, $\mathrm{WSL}^{\not\ni}[1\mathrm{G}]$) sentence $\phi \triangleq \wp\flat\psi \in \Phi$, we can always construct a PD-NWA $\mathcal{D}_{\psi}$ with $2^{\mathrm{O}(2^{|\psi|})}$ (*resp.*, $\mathrm{O}\left(2^{|\psi|^3}\right)$) states such that $\mathsf{L}(\mathcal{D}_{\psi}) = \mathsf{L}(\psi)$. By adding few more states, we can turn $\mathcal{D}_{\psi}$ into a PD-NWA $\mathcal{D}_{\widetilde{\phi}}$ recognising all models of the LTL (*resp.*, WLTL) formula $\widetilde{\phi} \triangleq \mathtt{G}\,(\ell(\phi) \rightarrow ((\mathtt{X}\,\mathtt{G}\,\neg\ell(\phi)) \wedge (\psi \vee \mathtt{F}\,\neg\flat)))$. This formula ensures that ψ is verified starting from the unique point where the corresponding atomic proposition $\ell(\phi)$ occurs, provided that binding $\flat$ is still active. By turning each PD-NWA $\mathcal{D}_{\widetilde{\phi}}$ into a tree automaton, we obtain the last component of $\mathcal{N}_{\varphi}$.

Construction 2 *(Sentence automaton). The* sentence automaton $\mathcal{N}_{\Phi}$ *is obtained as the product* $\prod_{\phi\in\Phi}\mathcal{N}_{\phi}$ *of the* NTAs $\mathrm{wot}_{\wedge}(\mathcal{D}_{\widetilde{\phi}})$ *derived from the* PD-NWA $\mathcal{D}_{\widetilde{\phi}}$*, for each* $\phi \in \Phi$.

At this point, it is clear that every tree accepted by $\mathcal{N}_{\varphi}$ is a normal model for the Skolem skeleton $\mathfrak{F}$ of φ, once the additional binding labelling is dropped. *Vice versa*, the underlying tree structure of a normal model is accepted by $\mathcal{N}_{\varphi}$ thanks to the fact that every $\mathcal{D}_{\widetilde{\phi}}$ is GFG for the class of bounded-fork trees, as shown in Theorem 12. This leads to the following result.

Theorem 13. *For every* $\mathrm{SL}^{\not\ni}_{k}[1\mathrm{G}]$ *(*resp.*,* $\mathrm{WSL}^{\not\ni}_{\kappa}[1\mathrm{G}]$*) sentence* φ*, there is a* k-NTA $\mathcal{N}_{\varphi}$ *of size* $2^{\mathrm{O}(2^{|\varphi|})}$ *(*resp.*,* $2^{|\varphi|^{\Theta(1)}}$*) recognising all and only the normal models of* φ.

By Theorem 8, we obtain that deciding $\mathrm{WSL}^{\not\ni}[1\mathrm{G}]$ is provably easier than deciding full SL[1G], known to be 2ExpTime-complete, while the complexity for $\mathrm{SL}^{\not\ni}[1\mathrm{G}]$ is lower if we assume the widely-shared conjecture that ExpSpace $\subset$ 2ExpTime. In more detail the ExpSpace bound for the $\mathrm{SL}^{\not\ni}_{k}[1\mathrm{G}]$ satisfiability problem is established by showing that the problems belongs to the class AExpTime([k-alt]) which is the class of problems solvable in exponential time by Alternating Turing Machines with at most k alternations between existential and universal quantifiers. We recall that the more general class of exponential-time bounded alternating Turing Machines with a polynomially bounded number of alternations is a relevant class included in ExpSpace which captures the precise complexity of some relevant problems, *e.g.*, the first-order theory of real addition with order [63]. While PSpace-hardness of $\mathrm{WSL}^{\not\ni}[1\mathrm{G}]$ trivially follows from an obvious encoding of standard modal-logic satisfiability, it is not known whether $\mathrm{SL}^{\not\ni}[1\mathrm{G}]$ is actually ExpTime-hard.

Theorem 14. $\mathrm{SL}^{\not\ni}_{k}[1\mathrm{G}]$ *(*resp.*,* $\mathrm{WSL}^{\not\ni}[1\mathrm{G}]$*) satisfiability problem is decidable in* AExpTime([k-alt]) *(*resp.*,* PSpace-complete*).*

Proof. Thanks to Theorem 8, the stated result immediately follows by solving on-the-fly the emptiness problem of the k-NTA $\mathcal{N}_{\varphi}$.

To see why all the steps involved in the satisfiability procedures, as well as their composition, can be carried out on-the-fly, we can make the following observations.

- Once the input automaton $\mathcal{N}_{\varphi}$ is provided on-the-fly, its emptiness can be checked on-the-fly as well.
- It is well-known that, given an LTL formula ψ, the corresponding equivalent NWA $\mathcal{W}_{\psi}$ can be constructed on-the-fly. The subset-like construction underlying the proof of Theorem 9 can also be computed on-the-fly. Hence, the PD-NWA $\mathcal{D}_{\psi}$ for ψ can be built on-the-fly. If we focus just on WLTL, it is immediate to observe that the PD-NWA $\mathcal{D}_{\psi}$ provided in the proof of Theorem 10 can also be built on-the-fly.
- Every NTA $\mathcal{N}_{\phi}$, for $\phi \in \Phi$, can be built on-the-fly, being a standard application of the wot function. Hence, the same holds for the product automaton $\mathcal{N}_{\Phi}$ as well.
- Finally, $\mathcal{N}_{\varphi} \triangleq \mathcal{D}_{\zeta} \times \mathcal{D}_{\mathfrak{F}} \times \mathcal{N}_{\Phi}$ is constructable on-the-fly, since the bounded-fork deterministic safety automaton $\mathcal{D}_{\mathfrak{F}}$, checking for the structural properties of the normal model, can obviously be provided on-the-fly to the product operation. □

Applying the same syntactic restrictions of Definition 5 to the syntax of ATL*, ATL, CTL*, and CTL, thus obtaining the non-recurrent fragments $\mathrm{ATL}^{*\not\ni}$, $\mathrm{ATL}^{\not\ni}$, $\mathrm{CTL}^{*\not\ni}$, and $\mathrm{CTL}^{\not\ni}$, one can leverage the result proved above to obtain improved complexity bounds for the satisfiability problem of these fragments.

Corollary 3. $\mathrm{ATL}^{*\emptyset}_{k}$ *and* $\mathrm{CTL}^{*\emptyset}_{k}$ *(*resp.*,* $\mathrm{ATL}^{\emptyset}$ *and* $\mathrm{CTL}^{\emptyset}$*) satisfiability problems are decidable in* AExpTime([k-alt]) *(*resp.*,* PSpace-complete*).*

7. Discussion

We have considered efficiently decidable fragments of One-Goal SL (SL[1G]), called *non-recurrent fragments*, where satisfaction requests for a sentence can only be iterated a bounded number of times along a computation. This is achieved by restricting the first (*resp.*, second) argument of the until (*resp.*, release) linear temporal operator. Specifically, when these arguments are limited to pure LTL formulae, we obtain that satisfiability is decidable in AExpTime([k-alt]) (which is known to be included in ExpSpace), where $k+1$ is the number of occurrences of principal subsentences within the formulae. If, however, those arguments are further restricted to Boolean formulae, completeness for PSpace is proven. Both non-recurrent fragments, which are strictly included in SL[1G], are still able to express non-trivial game-theoretic problems, such as the automatic synthesis of multi-agent systems, *e.g.*, communication protocols where active participants perform a bounded number of decisions during each communication session.

On the technical side, we obtain the complexity bounds by means of two main techniques. First, by exploiting a quasi-Herbrand property of a first-order characterisation of the sentences of those fragments, we identify a *normal-form* for the models of satisfiable sentences, which only admits a bounded number of branching point along any computation. Second, we leverage a novel class of automata, called *bounded-fork tree automata*, that can recognise normal models and whose emptiness problem can be decided in LogSpace.

Since SL[1G] strictly includes both ATL^{*} and CTL^{*}, the results immediately apply also to the corresponding non-recurrent fragments of those logics, where only LTL (*resp.*, Boolean) formulae can occur in the first (*resp.*, second) argument of an until (*resp.*, release) operator. In particular, this observation identifies novel fragments for both logics, namely the weak non-recurrent ones, with a satisfiability problem whose PSpace-complete complexity is strictly lower than the one for the full languages, known to be 2ExpTime-complete.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

Partially supported by the GNCS 2020 project “Ragionamento Strategico e Sintesi Automatica di Sistemi Multi-Agente”, GNCS 2022 project “Elaborazione del Linguaggio Naturale e Logica Temporale per la Formalizzazione di Testi”, and GNCS 2023 project “Analisi Simbolica e Numerica di Sistemi Ciberfisici”.

References

[1] A. Pnueli, The temporal logic of programs, in: Foundation of Computer Science'77, IEEE Computer Society, 1977, pp. 46–57.
[2] A. Pnueli, The temporal semantics of concurrent programs, Theor. Comput. Sci. 13 (1981) 45–60.
[3] A. Sistla, Theoretical Issues in the Design and Verification of Distributed Systems, Ph.D. thesis, Harvard University, Cambridge, MA, USA, 1983.
[4] A. Sistla, E. Clarke, The complexity of propositional linear temporal logics, in: Symposium on Theory of Computing'82, Association for Computing Machinery, 1982, pp. 159–168.
[5] A. Sistla, E. Clarke, The complexity of propositional linear temporal logics, J. ACM 32 (3) (1985) 733–749.
[6] M. Vardi, An automata-theoretic approach to linear temporal logic, in: Banff Higher Order Workshop'95, in: LNCS, vol. 1043, Springer, 1995, pp. 238–266.
[7] E. Emerson, E. Clarke, Design and synthesis of synchronization skeletons using branching-time temporal logic, in: Logic of Programs'81, in: LNCS, vol. 131, Springer, 1981, pp. 52–71.
[8] E. Emerson, E. Clarke, Design and synthesis of synchronization skeletons using branching-time temporal logic, Sci. Comput. Program. 2 (3) (1982) 241–266.
[9] E. Clarke, E. Emerson, A. Sistla, Automatic verification of finite-state concurrent systems using temporal logic specifications: a practical approach, in: Principles of Programming Languages'83, Association for Computing Machinery, 1983, pp. 117–126.
[10] E. Emerson, J. Halpern, “Sometimes” and “not never” revisited: on branching versus linear time, in: Principles of Programming Languages'83, Association for Computing Machinery, 1983, pp. 127–140.
[11] E. Clarke, E. Emerson, A. Sistla, Automatic verification of finite-state concurrent systems using temporal logic specifications, Trans. Program. Lang. Syst. 8 (2) (1986) 244–263.
[12] E. Emerson, J. Halpern, Decision procedures and expressiveness in the temporal logic of branching time, J. Comput. Syst. Sci. 30 (1) (1985) 1–24.
[13] E. Emerson, J. Halpern, “Sometimes” and “not never” revisited: on branching versus linear time, J. ACM 33 (1) (1986) 151–178.
[14] E. Emerson, C. Lei, Efficient model checking in fragments of the propositional muCalculus, in: Logic in Computer Science'86, IEEE Computer Society, 1986, pp. 267–278.
[15] E. Emerson, C. Jutla, A. Sistla, On model-checking for fragments of muCalculus, in: Computer Aided Verification'93, in: LNCS, vol. 697, Springer, 1993, pp. 385–396.
[16] A. Chandra, L. Stockmeyer, Alternation, in: Foundation of Computer Science'76, IEEE Computer Society, 1976, pp. 98–108.
[17] A. Chandra, D. Kozen, L. Stockmeyer, Alternation, J. ACM 28 (1) (1981) 114–133.
[18] S. Ghosh, R. Ramanujam, Strategies in games: a logic-automata study, in: European Summer School in Logic, Language, and Information'11, in: LNCS, vol. 7388, Springer, 2011, pp. 110–159.

[19] M. Benerecetti, F. Mogavero, A. Murano, Substructure temporal logic, in: Logic in Computer Science'13, IEEE Computer Society, 2013, pp. 368–377.
[20] J. Gutierrez, P. Harrenstein, M. Wooldridge, Iterated Boolean games, in: International Joint Conference on Artificial Intelligence'13, International Joint Conference on Artificial Intelligence' & AAAI Press, 2013, pp. 932–938.
[21] J. Gutierrez, P. Harrenstein, M. Wooldridge, Reasoning about equilibria in game-like concurrent systems, in: Knowledge Representation and Reasoning'14, AAAI Press, 2014, pp. 408–417.
[22] M. Benerecetti, F. Mogavero, A. Murano, Reasoning about substructures and games, Trans. Comput. Log. 16 (3) (2015), 25, pp. 1–46.
[23] J. Gutierrez, P. Harrenstein, M. Wooldridge, Iterated Boolean games, Inf. Comput. 242 (2015) 53–79.
[24] R. Alur, T. Henzinger, O. Kupferman, Alternating-time temporal logic, J. ACM 49 (5) (2002) 672–713.
[25] K. Chatterjee, T. Henzinger, N. Piterman, Strategy logic, in: Concurrency Theory'07, in: LNCS, vol. 4703, Springer, 2007, pp. 59–73.
[26] F. Mogavero, A. Murano, M. Vardi, Reasoning about strategies, in: Foundations of Software Technology and Theoretical Computer Science'10, in: LIPIcs, vol. 8, Leibniz-Zentrum fuer Informatik, 2010, pp. 133–144.
[27] K. Chatterjee, T. Henzinger, N. Piterman, Strategy logic, Inf. Comput. 208 (6) (2010) 677–693.
[28] F. Mogavero, A. Murano, G. Perelli, M. Vardi, Reasoning about strategies: on the model-checking problem, Trans. Comput. Log. 15 (4) (2014), 34, pp. 1–42.
[29] F. Mogavero, A. Murano, G. Perelli, M. Vardi, Reasoning about strategies: on the satisfiability problem, Log. Methods Comput. Sci. 13 (1:9) (2017) 1–37.
[30] P. Bouyer, P. Gardy, N. Markey, Weighted strategy logic with Boolean goals over one-counter games, in: Foundations of Software Technology and Theoretical Computer Science'15, in: LIPIcs, vol. 45, Leibniz-Zentrum fuer Informatik, 2015, pp. 69–83.
[31] P. Bouyer, P. Gardy, N. Markey, On the semantics of strategy logic, Inf. Process. Lett. 116 (2) (2016) 75–79.
[32] P. Gardy, P. Bouyer, N. Markey, Dependences in strategy logic, in: Symposium on Theoretical Aspects of Computer Science'18, in: LIPIcs, vol. 96, Leibniz-Zentrum fuer Informatik, 2018, pp. 34:1–15.
[33] P. Gardy, P. Bouyer, N. Markey, Dependences in strategy logic, Theor. Comput. Sci. 64 (3) (2020) 467–507.
[34] F. Laroussinie, N. Markey, Satisfiability of ATL with strategy contexts, in: Games, Automata, Logics, and Formal Verification'13, in: EPTCS, vol. 119, 2013, pp. 208–223.
[35] F. Laroussinie, N. Markey, Augmenting ATL with strategy contexts, Inf. Comput. 245 (2015) 98–123.
[36] F. Mogavero, A. Murano, G. Perelli, M. Vardi, What makes ATL* decidable? A decidable fragment of strategy logic, in: Concurrency Theory'12, in: LNCS, vol. 7454, Springer, 2012, pp. 193–208.
[37] E. Acar, M. Benerecetti, F. Mogavero, Satisfiability in strategy logic can be easier than model checking, in: AAAI Press19, AAAI Press, 2019, pp. 2638–2645.
[38] F. Mogavero, G. Perelli, Binding forms in first-order logic, in: Computer Science Logic'15, in: LIPIcs, vol. 41, Leibniz-Zentrum fuer Informatik, 2015, pp. 648–665.
[39] S. Bova, F. Mogavero, Herbrand property, finite quasi-herbrand models, and a Chandra-Merlin theorem for quantified conjunctive queries, in: Logic in Computer Science'17, Association for Computing Machinery, 2017, pp. 1–12.
[40] D. Dams, Flat fragments of CTL and CTL*: separating the expressive and distinguishing powers, Log. J. IGPL 7 (1) (1999) 55–78.
[41] H. Comon, V. Cortier, Flatness is not a weakness, in: Computer Science Logic'00, in: LNCS, vol. 1862, Springer, 2000, pp. 262–276.
[42] S. Demri, R. Lazic, D. Nowak, On the freeze quantifier in constraint LTL: decidability and complexity, Inf. Comput. 205 (1) (2007) 2–24.
[43] O. Ibarra, Z. Dang, On removing the pushdown stack in reachability constructions, in: International Symposium on Algorithms and Computation'01, in: LNCS, vol. 2223, Springer, 2001, pp. 244–256.
[44] G. Fainekos, S. Loizou, G. Pappas, Translating temporal logic to controller specifications, in: Decision and Control'06, IEEE Computer Society, 2006, pp. 899–904.
[45] S. Demri, P. Schnoebelen, The complexity of propositional linear temporal logics in simple cases, Inf. Comput. 174 (1) (2002) 84–103.
[46] P. Schnoebelen, The complexity of temporal logic model checking, in: Advances in Modal Logic'02, College Publications, 2002, pp. 393–436.
[47] B. Khoussainov, A. Nerode, Automata Theory and Its Applications, Birkhauser, 2001.
[48] E. Grädel, W. Thomas, T. Wilke, Automata, Logics, and Infinite Games: A Guide to Current Research, LNCS, vol. 2500, Springer, 2002.
[49] F. Baader, W. Snyder, Unification theory, in: Handbook of Automated Reasoning, vol. 1, Elsevier & MIT Press, 2001, pp. 445–532.
[50] F. Mogavero, A. Murano, M. Vardi, Relentful strategic reasoning in alternating-time temporal logic, in: Logic for Programming Artificial Intelligence and Reasoning'10, in: LNAI, vol. 6355, Springer, 2010, pp. 371–387.
[51] O. Kupferman, M. Vardi, P. Wolper, An automata theoretic approach to branching-time model checking, J. ACM 47 (2) (2000) 312–360.
[52] E. Amparore, S. Donatelli, F. Gallà, A CTL* model checker for Petri nets, in: Application and Theory of Petri Nets and Concurrency'20, in: LNCS, vol. 12152, Springer, 2020, pp. 403–413.
[53] S. Schewe, ATL* satisfiability is 2ExpTime-complete, in: International Colloquium on Automata, Languages, and Programming'08, in: LNCS, vol. 5126, Springer, 2008, pp. 373–385.
[54] M. Vardi, P. Wolper, Automata-theoretic techniques for modal logics of programs, J. Comput. Syst. Sci. 32 (2) (1986) 183–221.
[55] R. McNaughton, Infinite games played on finite graphs, Ann. Pure Appl. Log. 65 (1993) 149–184.
[56] L. Stockmeyer, The polynomial-time hierarchy, Theor. Comput. Sci. 3 (1) (1976) 1–22.
[57] C. Papadimitriou, Computational Complexity, Addison-Wesley, 1994.
[58] A. Pnueli, R. Rosner, On the synthesis of a reactive module, in: Principles of Programming Languages'89, Association for Computing Machinery, 1989, pp. 179–190.
[59] M. Vardi, Reasoning about the past with two-way automata, in: International Colloquium on Automata, Languages, and Programming'98, in: LNCS, vol. 1443, Springer, 1998, pp. 628–641.
[60] T. Henzinger, N. Piterman, Solving games without determinization, in: Computer Science Logic'06, in: LNCS, vol. 4207, Springer, 2006, pp. 395–410.
[61] M. Vardi, P. Wolper, An automata-theoretic approach to automatic program verification, in: Logic in Computer Science'86, IEEE Computer Society, 1986, pp. 332–344.
[62] C. Löding, Methods for the Transformation of ω-Automata: Complexity and Connection to Second Order Logic, Master's thesis, Kiel University, Kiel, Germany, 1998.
[63] J. Ferrante, C. Rackoff, A decision procedure for the first order theory of real addition with order, SIAM J. Comput. 4 (1) (1975) 69–76.