Taming Strategy Logic: Non-Recurrent Fragments

Massimo Benerecetti, Fabio Mogavero, and Adriano Peron ✉
Università degli Studi di Napoli Federico II

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 have 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 logics, and for the corresponding fragments of ATL* and CTL*, is ExpSpace and PSpace-complete, respectively.

2012 ACM Subject Classification Theory of computation → Modal and temporal logics; Theory of computation → Logic and verification; Theory of computation → Automata over infinite objects

Keywords and phrases Strategic Reasoning, Multi-Agent Systems, Temporal Logics, Satisfiability.

Digital Object Identifier 10.4230/LIPIcs.TIME.2022.14

1 Introduction

A number of extensions of temporal logics specifically tailored to 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 [22, 5, 24, 25, 6, 26]. *Alternating-Time Temporal Logic* (ATL*, for short) was originally introduced by Alur, Henzinger, and Kupferman [2] and allows for reasoning about *strategic behaviour* of agents with temporal goals. This logic generalises the branching-time temporal logic CTL* [17, 18] 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) [10, 34, 11, 32, 33], instead, extends LTL by means of 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 respectively read 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 (see [32, 33]).

The *model-checking problem* for SL and for many of its fragments has been studied with some depth and is relatively well-understood [32, 7, 8, 20, 21]. The picture is, however, much less clear when *satisfiability* is considered. The full logic SL is known to be undecidable [34]. The *one-goal* fragment (SL[1G], for short), where only a single binding prefix is allowed in any sentence, is decidable in 2ExpTime [31]. 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 [33]. Recently, the *flat* fragment of *conjunctive-goal* SL has been studied in [1], which provides a PSPACE-COMPLETE result for the problem, witnessing the quite 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$^{\not\ni}$[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$^{\not\ni}$[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 [35, 9]. Then, we introduce a novel class of tree automata, called *bounded-fork automata*, accepting trees with bounded-branching. 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 [13] 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 [12, 14], analysis of discrete pushdown timed systems [28], and the synthesis of hybrid systems [19]. In particular, the LTL fragment considered in [12, 28] is a sublogic of the linear-time logic underlying WSL$^{\not\ni}$[1G], while the one originally considered in [13] 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 [15, 38], to the best of our knowledge, only expressiveness properties have been studied for flatCTL and flatCTL*.

2 Preliminaries

Games. A *concurrent game structure* (CGS, for short) *w.r.t.* finite non-empty 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* $\pi \in \mathrm{Pth} \subseteq \mathrm{Ps}^\infty \triangleq \mathrm{Ps}^* \cup \mathrm{Ps}^\omega$ is a finite or infinite sequence of positions compatible with the move function, *i.e.*, $((\pi)_i, (\pi)_{i+1}) \in \tau$, for each $i \in [0, |\pi| - 1)$. The set $\mathrm{Pth}(v) \triangleq \{\pi \in \mathrm{Pth} \mid |\pi| > 0 \wedge \mathsf{fst}(\pi) = v\}$ denotes the set of paths starting at a position v. A *history* at v is a finite non-empty 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 $a \in \mathrm{Ag}$. 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 all position $v, w \in \mathrm{Ps}$, for some $x \in \mathrm{X}$. 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, $|\{i \in \mathbb{N} \mid |\tau((\pi)_i)| > 1\}| \leq k$.

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 over $\mathfrak{F}$ 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 [4, 9].

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 finite non-empty 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*, and $\delta\colon \mathrm{Q} \times \Sigma \to \mathrm{Q} \cup \{\bot, \top\}$ (*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) $\delta(q,\sigma)$. 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 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 $\delta(q,\sigma,d) \subseteq \mathrm{Q}^d$), where $\bot$ 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). We only consider the Büchi acceptance condition, for both word and tree automata. The notions of *(accepting) run* and *recognised*

language are the standard ones. For more details, we refer to [29, 23].

3 Decidable Fragments of Strategy Logic

Strategy Logic [10, 34] extends LTL by allowing to *quantify* over strategies and to assign a strategy to each agent, by *binding* the latter with some quantified variable. A *quantifier prefix* is a finite sequence $\wp \in \mathrm{Qn} \subseteq \{\exists x, \forall x \mid x \in \mathrm{Vr}\}^*$ of existential and universal quantifiers $\mathtt{Qn}x$, in which variables $x \in \mathrm{Vr}$ occur at most once. Similarly, a *binding prefix* is a finite sequence $\flat \in \mathrm{Bn} \subseteq (\mathrm{Ag} \times \mathrm{Vr})^{|\mathrm{Ag}|}$ of bindings (a, x), in which each agent $a \in \mathrm{Ag}$ occurs exactly once. By $\mathsf{vr}(\wp), \mathsf{vr}(\flat) \subseteq \mathrm{Vr}$ we denote the sets of variables occurring in $\wp$ and $\flat$.

One-Goal Strategy Logic. *One-Goal Strategy Logic* is one of the largest decidable fragments of SL known to date and is complete for 2ExpTime. 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 them again in a nested subsentence.

▶ **Definition 1** (SL[1G] Syntax [31])**.** SL[1G] formulas *are generated from the sets of atomic propositions* AP*, quantifier prefixes* Qn*, and binding prefixes* Bn *via the following grammar, where* $p \in \mathrm{AP}$*,* $\wp \in \mathrm{Qn}$*, and* $\flat \in \mathrm{Bn}$*, with* $\mathsf{vr}(\wp) = \mathsf{vr}(\flat)$*:*

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

SL[1G] *denotes the set of sentences generated by Rule* φ*, while* FSL[1G] $\subset$ SL[1G] *represents the* flat fragment*,* i.e.*, the subset generated by the variant of the grammar where* p *replaces the call to Rule* φ *within Rule* ψ*,* i.e.*, with* ψ *pure* LTL*.*

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 φ. 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, to take care of bindings assigning strategies to agents. Let $\mathrm{Asg}(v) \triangleq (\mathrm{Vr} \cup \mathrm{Ag}) \rightharpoonup \mathrm{Str}(v)$ denote the set of such assignments. For a set $\mathrm{V} \subseteq (\mathrm{Vr} \cup \mathrm{Ag})$, we also provide, for convenience, the set of assignments defined only 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. Since the semantics for the Boolean and temporal operators is practically the classic one (see [32]), we only provide the interpretation of quantifiers and bindings.

▶ **Definition 2** (SL Semantics [32])**.** *Given a* CGS $\mathfrak{G}$*, for all* SL *formulas* φ*, positions* $v \in \mathrm{Ps}$*, and* v*-rooted assignments* $\chi \in \mathrm{Asg}_{\subseteq}(v, \mathsf{free}(\varphi))$*, the modelling relation* $\mathfrak{G}, v, \chi \models \varphi$ *is inductively defined as follows.*

1. *Atomic propositions, Boolean connectives and temporal operators are interpreted as usual.*
2. *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* $\sigma \in \mathrm{Str}(v)$*.*
3. *For all* $a \in \mathrm{Ag}$ *and* $x \in \mathrm{Vr}$*:* $\mathfrak{G}, v, \chi \models (a, x)\phi$*, if* $\mathfrak{G}, v, \chi[a \mapsto \chi(x)] \models \phi$*.*

For a sentence φ, we write $\mathfrak{G}, v \models \varphi$ and $\mathfrak{G} \models \varphi$ instead of $\mathfrak{G}, v, \varnothing \models \varphi$ and $\mathfrak{G}, v_I, \varnothing \models \varphi$.

The existence of a normal model for 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 a reminiscent of the technique used in the model-checking algorithms for CTL* [30, 3].

For example, the following sentence $\varphi \triangleq p \wedge \forall x \exists y \forall z (a,x)(b,y)(c,z)(\mathtt{X}\,(q \wedge (\mathtt{X}\,\mathtt{F}\,q)\,\mathtt{U}\,(\phi_1 \wedge \phi_2)))$ with $\phi_1 \triangleq \exists x \forall y (a,x)(b,y)(c,y)(p\,\mathtt{U}\,q)$ and $\phi_2 \triangleq \forall x \exists y (a,y)(b,x)(c,y)(\mathtt{G}\,\neg q)$ can be stratified into 2 layers, using the fresh atomic propositions $\{s, s_1, s_2\}$ as names for the subsentences of φ: $\phi_1 \mapsto s_1$, $\phi_2 \mapsto s_2$ and $\forall x \exists y \forall z (a,x)(b,y)(c,z)(\mathtt{X}\,(q \wedge (\mathtt{X}\,\mathtt{F}\,q)\,\mathtt{U}\,(s_1 \wedge s_2))) \mapsto s$. In the end, the original formula φ is summarised by the positive Boolean formula $\zeta \triangleq p \wedge s$. This idea 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 BF (*resp.*, BF^+) the set of Boolean (*resp.*, positive Boolean) 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] *sentences, and* $\ell \colon \Phi \to \mathrm{AP}$ *is an injective function mapping each sentence* $\phi \in \Phi$ *to an atomic proposition* $\ell(\phi) \in \mathrm{AP}$*, for which there is a strict partial order* $\prec \subseteq \Phi \times \Phi$ *such that if* $\ell(\phi) \in \mathsf{ap}(\phi')$ *then* $\phi \prec \phi'$*, for all* $\phi' \in \Phi$*. In addition:* $\mathfrak{d}$ *is* simple *if every atomic proposition* $p \in \mathsf{img}(\ell)$ *occurs in exactly one sentence* $\phi \in \Phi \cup \{\zeta\}$ *and at most once in it;* $\mathfrak{d}$ *is* principal *if it is simple and all sentences* ϕ *in* Φ *have the form* $\wp \flat \psi$*, for some* $\wp \in \mathrm{Qn}$*,* $\flat \in \mathrm{Bn}$*, and* $\psi \in \mathrm{LTL}$*.*

For instance, the skeleton of the example above is indeed a principal one. 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. While, for convenience, we allow for more liberal forms of skeletons, principal ones suffice, as one such skeleton exists for each sentence, where different occurrences of the same subsentence are mapped to different names by ℓ.

▶ **Proposition 4.** *Each* SL[1G] *sentence* φ *enjoys a principal* SL[1G] *skeleton* $\mathfrak{d}$ *with* $\varphi = \varphi_{\mathfrak{d}}$*.*

Satisfaction of a skeleton $\mathfrak{d}$ by a CGS $\mathfrak{G}$ over the atomic propositions of $\mathfrak{d}$ is defined quite naturally. Specifically, the initial position of $\mathfrak{G}$ must satisfy locally 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 5** (Skeleton Satisfaction). *A* CGS $\mathfrak{G}$ *satisfies an* SL[1G] *skeleton* $\mathfrak{d}$*, in symbols* $\mathfrak{G} \models \mathfrak{d}$*, 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.

▶ **Theorem 6.** $\varphi_{\mathfrak{d}}$ *is satisfiable* iff $\mathfrak{d}$ *is satisfied by a tree* CGS*, for every* SL[1G] *skeleton* $\mathfrak{d}$*.*

Non-Recurrent One-Goal Strategy Logics. The main source of complexity for SL[1G], or CTL* and ATL* for that matter, resides in its 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 $\mathtt{EG}\,(\neg p \wedge \mathtt{EX}\,p)$. Given the branching nature of quantifications in SL, this may lead to models with an unbounded number of branching points. In general, such models can be recognised by nondeterministic tree automata with a double exponential

number of states [37, 31]. Emptiness for such tree automata is known to be PTime [42], which leads to a 2ExpTime procedure for deciding SL[1G]. Since CTL* is contained in SL[1G], completeness for 2ExpTime follows [16]. 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.

▶ **Definition 7** (SL[1G] Fragments). *Formulas of* non-recurrent fragments *of* SL[1G] *are generated from the sets of atomic propositions* AP*, quantifier prefixes* Qn*, and binding prefixes* Bn *via the following grammar, with* $p \in \mathrm{AP}$*,* $\psi \in \mathrm{LTL}(\mathrm{AP})$*,* $\beta \in \mathrm{BF}(\mathrm{AP})$*,* $\wp \in \mathrm{Qn}$*, and* $\flat \in \mathrm{Bn}$ *such that* $\mathsf{vr}(\wp) = \mathsf{vr}(\flat)$*:*

$$\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]\colon \quad \varphi := p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \wp \flat \eta \mid \wp \flat \psi; \qquad \eta := \varphi \mid \eta \wedge \eta \mid \eta \vee \eta \mid \psi \,\mathtt{U}\, \varphi \mid \varphi \,\mathtt{R}\, \psi \mid \mathtt{X}\eta \mid \mathtt{X}\psi;$$
$$\mathrm{WSL}^{\not\emptyset}[1\mathrm{G}]\colon \quad \varphi := p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \wp \flat \eta; \qquad \eta := \varphi \mid \eta \wedge \eta \mid \eta \vee \eta \mid \beta \,\mathtt{U}\, \varphi \mid \varphi \,\mathtt{R}\, \beta \mid \mathtt{X}\eta;$$

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. By replacing all the occurrences of φ in the two rules η with a positive Boolean formula $\gamma \in \mathrm{BF}^{+}(\mathrm{AP} \cup \mathrm{A})$, for a (possibly empty) distinguished set A of atomic propositions, such that $\mathrm{AP} \cap \mathrm{A} = \emptyset$, we obtain the corresponding flat fragments $\mathrm{FSL}^{\not\emptyset}_{\mathrm{A}}[1\mathrm{G}]$ and $\mathrm{FWSL}^{\not\emptyset}_{\mathrm{A}}[1\mathrm{G}]$. The idea is that A contains names of sentences possibly used by the skeletons for the two fragments. In addition, we call *Weak* LTL (WLTL for short), the fragment of LTL that agrees with the rule η of $\mathrm{FWSL}^{\not\emptyset}[1\mathrm{G}]$.

We can obtain skeletons for the new fragments, by suitably restricting their components to the corresponding flat fragments and requiring that only fresh atomic propositions in A be used as names for sentences. An $\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]$ (*resp.*, $\mathrm{WSL}^{\not\emptyset}[1\mathrm{G}]$) skeleton $\mathfrak{d} = \langle \zeta, \Phi, \ell \rangle$ is a principal SL[1G] skeleton such that *1)* $\Phi \subseteq \mathrm{FSL}^{\not\emptyset}_{\mathrm{A}}[1\mathrm{G}]$ (*resp.*, $\Phi \subseteq \mathrm{FWSL}^{\not\emptyset}_{\mathrm{A}}[1\mathrm{G}]$), and *2)* $\mathsf{img}(\ell) \cap \mathrm{A} = \emptyset$, for some $\mathrm{A} \subseteq \mathrm{AP}$. The analogous of Proposition 4 holds for the two new fragments $\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]$ and $\mathrm{WSL}^{\not\emptyset}[1\mathrm{G}]$ as well.

▶ **Proposition 8.** *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* $\mathfrak{d}$ *with* $\varphi = \varphi_{\mathfrak{d}}$*.*

The constraint on the non-recurrence of sentences allows us to strengthen Theorem 6 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 9** (Single-Time Skeleton Satisfaction). *A* CGS $\mathfrak{G}$ single-time satisfies *a skeleton* $\mathfrak{d}$ *if* 1) $\mathfrak{G} \models \mathfrak{d}$ *and* 2) *if* $\ell(\phi) \in \lambda(v)$ *then* $\ell(\phi) \notin \lambda(w)$*, for all* $\phi \in \Phi$*,* $v \in \mathrm{Ps}$*, and* $w \in \tau^{+}(v)$*.*

▶ **Theorem 10.** $\varphi_{\mathfrak{d}}$ *is satisfiable* iff $\mathfrak{d}$ *is single-time satisfied by a tree* CGS*, for every* $\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]$ *skeleton* $\mathfrak{d}$*.*

Assume $\varphi_{\eth}$ is satisfiable. By Theorem 6, $\eth$ 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 $\mathfrak{G}_{k+1}, \mathfrak{G}_k, \mathfrak{G}_{k-1}, \ldots, \mathfrak{G}_0$ 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 $i < k$ just need to 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 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 $\eth$.

4 Normal Models

The efficient satisfiability of 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, to allow for bindings containing strategy terms, instead of simple variables, which enables us to state a Skolem normal-form theorem for SL[1G]. This result can be used to show that any model for a sentence φ of $\mathrm{SL}^{\varnothing}[1\mathrm{G}]$ 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.

Functions in SL. Given a function signature $\mathcal{F}$, by SL[1G,$\mathcal{F}$] we denote the extension of SL[1G], where we allow agents to be bound with complex terms instead of 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. $\forall$SL[1G,$\mathcal{F}$] represents the universal fragment of SL[1G,$\mathcal{F}$], where existential quantifiers are forbidden. In order to define the semantics of an SL[1G,$\mathcal{F}$] sentence, we need to provide a *strategy interpretation* for all function symbols in Fn. We do this, via the map $\Im\colon v \in \mathrm{Ps} \mapsto \mathfrak{F}_v$ assigning to each position v an $\mathcal{F}$-structure $\mathfrak{F}_v = \langle \mathrm{Str}(v), \cdot^{\Im_v} \rangle$ whose domain 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 modelling relation $(\mathfrak{G}, \Im), v, \chi \models \varphi$ as in Definition 2, where Item 3 gets replaced by the following one:

- for all $a \in \mathrm{Ag}$ and $t \in \mathrm{Tr}$: $(\mathfrak{G}, \Im), v, \chi \models (a, t)\phi$, if $(\mathfrak{G}, \Im), v, \chi[a \mapsto t^{\Im(v),\chi}] \models \phi$,

where agent $a \in \mathrm{Ag}$ is bound to strategy $t^{\Im(v),\chi}$, *i.e.*, the interpretation of 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 agent a a strategy dependent on those associated with the variables occurring in the term t.

An SL[1G,$\mathcal{F}$] 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, by $\mathsf{skm}\colon \mathrm{SL}[1\mathrm{G}] \to \forall\mathrm{SL}[1\mathrm{G},\mathcal{F}]$ 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. As an example, consider the SL[1G] sentence φ used in the previous section to exemplify the notion of SL[1G] skeleton.

Then, $\mathsf{skm}(\varphi) = p \wedge \forall x \forall z (a,x)(b,f_1(x))(c,z)(\mathtt{X}\,(q \wedge (\mathtt{X}\,\mathtt{F}\,q)\,\mathtt{U}\,(\mathsf{skm}(\phi_1) \wedge \mathsf{skm}(\phi_2))))$, where $\mathsf{skm}(\phi_1) = \forall y (a,f_2)(b,y)(c,y)(p\,\mathtt{U}\,q)$ and $\mathsf{skm}(\phi_2) = \forall x (a,f_3(x))(b,x)(c,f_3(x))(\mathtt{G}\,\neg q)$. 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 [32] (see Theorem 4.5 and Corollary 4.6), it has been proved that SL enjoys a semantic version of *Skolem normal-form theorem*, where the interpretation of *Skolem functions* 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.

▶ **Theorem 11.** *Let* $\mathfrak{G}$ *be a* CGS. *An* SL[1G] *sentence* φ *is satisfied by* $\mathfrak{G}$ iff *the* $\forall$SL[1G,$\mathcal{F}$] *sentence* $\mathsf{skm}(\varphi)$ *is satisfied by* $\mathfrak{G}$.

A fundamental property of SL[1G], which allows both its model-checking and satisfiability problem to be elementary decidable [32, 31, 33], is that every satisfiable sentence of this logic is *behaviourally satisfiable* [32] (see Theorem 4.20 and Corollary 4.21), with the intuitive meaning 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 proved in [32] as follows. We say that two strategies $\sigma_1, \sigma_2 \in \mathrm{Str}(v)$ are *equal along* history $\rho \in \mathrm{Hst}(v)$ (ρ-*equal*, for short), if, for every history $\rho' \in \mathrm{Hst}(v)$ with $\rho' \leq \rho$, it holds that $\sigma_1(\rho') = \sigma_2(\rho')$, where $\leq$ is the partial order induced by prefixes. This notion immediately lifts to vectors of strategies $\vec{\sigma}_1, \vec{\sigma}_2 \in \mathrm{Str}(v)^k$, of 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 $i \in [k]$. A function between strategies $\mathsf{f} \colon \mathrm{Str}(v)^k \to \mathrm{Str}(v)$, for some dimension $k \in \mathbb{N}$, 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 $\mathsf{f}(\vec{\sigma}_1)(\rho) = \mathsf{f}(\vec{\sigma}_2)(\rho)$. 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 SL[1G,$\mathcal{F}$] sentence φ is *behaviourally satisfied* by a CGS $\mathfrak{G}$ if there exists a behavioural strategy interpretation $\Im$ such that $(\mathfrak{G}, \Im) \models \varphi$.

▶ **Theorem 12.** *For any* CGS $\mathfrak{G}$ *and* SL[1G] *sentence* φ, *the sentence* $\mathsf{skm}(\varphi)$ *is satisfied by* $\mathfrak{G}$ iff *it is behaviourally satisfied by* $\mathfrak{G}$.

Unifying Bindings & Paths. In [35] it has been observed that the decidability of the satisfiability problem for SL[1G] can be attributed to the fact that variables are indivisibly associated with agents by bindings. Here we further exploit that observation to define a normal form for SL$^{\varnothing}$[1G] models, applying the notions of *Herbrand property* and *quasi-Herbrand structures* devised in [9], so that unifying bindings identify the same paths.

The notion of SL[1G] (*resp.*, SL$^{\varnothing}$[1G]) skeleton, as well as the corresponding concept of (*resp.*, single-time) skeleton satisfaction, immediately lifts to SL[1G,$\mathcal{F}$] (*resp.*, SL$^{\varnothing}$[1G,$\mathcal{F}$]) in the obvious way. A skeleton is universal if all formulas in Φ are universal, *i.e.*, $\Phi \subseteq \forall$SL[1G,$\mathcal{F}$]. Given an SL[1G] (*resp.*, SL$^{\varnothing}$[1G], WSL$^{\varnothing}$[1G]) skeleton $\eth$, we denote by $\mathsf{skm}(\eth)$ the (universal) $\forall$SL[1G,$\mathcal{F}$] (*resp.*, $\forall$SL$^{\varnothing}$[1G,$\mathcal{F}$], $\forall$WSL$^{\varnothing}$[1G,$\mathcal{F}$]) skeleton obtained via Skolemisation of all the sentences in Φ, where a different set of Skolem symbols is used for each sentence.

The following result is an easy corollary of what we have derived. Indeed, Theorem 10 ensures that, for every SL$^{\varnothing}$[1G] skeleton $\eth$, the sentence $\varphi_{\eth}$ is satisfiable *iff* $\eth$ is single-time satisfiable by some tree CGS $\mathfrak{G}$. Now, by Theorem 11, $\mathfrak{G}, v \models \phi$ *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 12 allows for a behavioural satisfaction.

▶ **Corollary 13.** *For every* $\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]$ *skeleton* $\mathfrak{d}$*, it holds that* $\varphi_{\mathfrak{d}}$ *is satisfiable* iff $\mathsf{skm}(\mathfrak{d})$ *is single-time behaviourally satisfiable by a tree* CGS*.*

Bindings $\flat = (a_1, t_1) \cdots (a_k, t_k) \in \mathrm{Bn}(\mathcal{F})$ are sequences of terms over $\mathcal{F}$, 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 bindings $\mathrm{B} \subseteq \mathrm{Bn}(\mathcal{F})$ *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* over $\mathfrak{F}$ if there is an assignment χ such that $\flat_1^{\mathfrak{F},\chi} = \flat_2^{\mathfrak{F},\chi}$, for all $\flat_1, \flat_2 \in \mathrm{B}$. As an example, consider the bindings $\flat_1 \triangleq (a, x)(b, f_1(x))(c, z)$, $\flat_2 \triangleq (a, f_2)(b, y)(c, y)$, and $\flat_3 \triangleq (a, f_3(x))(b, x)(c, f_3(x))$, previously obtained by Skolemisation. One can see that $\flat_1$ and $\flat_2$ unify in $(a, f_2)(b, f_1(f_2))(c, f_1(f_2))$, while neither $\flat_1$ and $\flat_3$ nor $\flat_2$ and $\flat_3$ unify. By a result in [9] (see Theorem 1) $\flat_1$ and $\flat_2$ also equalise over 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 [9]) over which $\flat_3$ does not equalise with either $\flat_1$ or $\flat_2$.

Every finite set of bindings $\mathrm{B} \subset \mathrm{Bn}(\mathcal{F})$ is associated with its *maximally unifiable coverage* $\mathsf{muc}(\mathrm{B}) \subseteq 2^{\mathrm{B}}$, *i.e.*, the unique set of the 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 $\flat \in \mathrm{B} \setminus \mathrm{C}$. As an example, consider the set of three bindings $\mathrm{B} \triangleq \{\flat_4, \flat_5, \flat_6\}$, 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 bindings 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 bindings unify in $\flat_{56} \triangleq (\alpha, w)(\beta, f(w))(\gamma, g(f(w)))$. In addition, the whole 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 bindings $\{\flat_1, \flat_2, \flat_3\}$ of the previous paragraph, we have that $\mathsf{muc}(\{\flat_1, \flat_2, \flat_3\}) = \{\{\flat_1, \flat_2\}, \{\flat_3\}\}$.

A *normal model* of a universal skeleton $\mathfrak{d}$ is an interpreted tree CGS $(\mathfrak{G}, \mathfrak{I})$, where the number $|\tau(v)|$ of successors of each position $v \in \mathrm{Ps}$ is dictated solely by the set of bindings $\flat$ of some sentence $\phi \in \Phi$, whose induced play $\pi = \mathsf{play}(\flat^{\mathfrak{I}(w),\chi}, w)$, with $\chi \in \mathrm{Asg}(w)$ 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 bindings, 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 $\forall \flat_2 \psi_2$. This witnesses that both sentences must be satisfied at v. If bindings $\flat_1$ and $\flat_2$ unify, hence equalise, then the corresponding LTL matrices ψ_1 and ψ_2 must necessarily be satisfied along the same paths from v, as the two bindings induce the same paths. 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 bindings can have different interpretations. Normal models capture this intuition, by keeping track, at each position, of which bindings are paired with which paths from that position. To this end, such models are equipped with three functions: a *global binding* function g that associates with each position v the set of bindings paired with all the paths through v; a *local binding* function l, associating with each position v the set of bindings of the sentences that label v, *i.e.* the sentences that must be satisfied starting from v; and a *routing* map r that, based on (non)unification of the bindings at v, dispatches them along possibly different paths from v.

▶ **Definition 14** (Normal Model)**.** *An interpreted* CGS $(\mathfrak{G}, \mathfrak{I})$ *satisfying a* $\forall\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]$ *skeleton* $\mathfrak{d}$ *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* $\mathsf{r} \colon v \in \mathrm{Ps} \mapsto$

$(\tau(v) \to \mathsf{muc}(\mathsf{g}(v)))$ *enjoying the following properties, for all positions* $v \in \mathrm{Ps}$*:*

a) $\mathsf{r}(v)$ *is a bijective map from* $\tau(v)$ *to* $\mathsf{muc}(\mathsf{g}(v))$*;*

b) $\mathsf{l}(v) = \left\{ \flat \in \mathrm{Bn} \;\middle|\; \exists \phi \triangleq \forall \flat \psi \in \Phi.\, \ell(\phi) \in \lambda(v) \right\}$*;*

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* $\flat \in \mathsf{l}(v)$*,* $\chi \in \mathrm{Asg}(v, \mathsf{vr}(\flat))$*, and* $i \in \mathbb{N}$*, where* $\pi \triangleq \mathsf{play}(\flat^{\Im(v),\chi}, v)$*.*

For each position v, by Item b, the local binding function l identifies the set of bindings of those universal sentences $\phi \in \Phi$ whose atomic proposition $\ell(\phi)$ labels v (note that ϕ holds at v due to Item 2 of Definition 5); by Item c, the global binding function g extends l with the bindings of the sentences satisfied at some ancestor of v; finally, by Item a, the routing map r distributes the bindings collected by g across the successors of v, in such a way that all bindings forming a maximally unifiable set are routed towards the same successor, while different unifying sets are routed towards different successors. Observe that, due to Item d, a path induced by a binding $\flat$ necessarily passes through one of the successors chosen by r for $\flat$ and, *vice versa*, a successor chosen for $\flat$ is traversed by at least one path induced by $\flat$. Hence, Item d captures the requirement that, at each position, normal models keep track of which bindings are paired with which paths from that position.

Thanks to Corollary 13 and the notion of behavioural satisfaction, from the strategy interpretation $\Im(v) = \langle \mathrm{Str}(v), \cdot^{\Im(v)} \rangle$ at each position $v \in \mathrm{Ps}$ of a tree CGS $\mathfrak{G}$ one can extract infinitely-many action interpretations $\mathfrak{F}^v_\rho = \left\langle \mathrm{Ac}, \cdot^{\mathfrak{F}^v_\rho} \right\rangle$, one for each history $\rho \in \mathrm{Hst}(v)$ starting at v in $\mathfrak{G}$. 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-tuple of strategies $\vec{\sigma} \in \mathrm{Str}(v)^k$, where the i-th element $(\vec{c})_i$ of $\vec{c}$ is equal to the action $(\vec{\sigma})_i(\rho)$ chosen by the i-th strategy $(\vec{\sigma})_i$ of $\vec{\sigma}$ at ρ. In a sense, 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. By exploiting the connection between strategy and action interpretations and using the fact that, for each set of bindings $\mathrm{B} \subseteq \mathrm{Bn}(\mathcal{F})$, there is always an $\mathcal{F}$-structure $\mathfrak{F}^{\star v}_\rho$ for which every $\mathrm{X} \subseteq \mathrm{B}$ unifies *iff* X equalises over $\mathfrak{F}^{\star v}_\rho$ (see Theorem 2 of [9]), the following result can be obtained.

▶ **Theorem 15.** *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 satisfiable.*

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: *i)* 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; *ii)* thanks to the normality property, a fork at any given position v of a path is only caused by non-unifying bindings, which occur if new sentences in Φ need to be satisfied at v, as the bindings routed toward v from the ancestors necessarily unify.

▶ **Corollary 16.** *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* $k \triangleq \mathsf{max}\{0, |\Phi| - 1\}$*.*

5 New Classes of Automata

In this section, we introduce the novel class of *bounded-fork tree automata* that will be exploited in Section 6 to devise an efficient satisfiability checking algorithm for the non-recurrent fragments of $\mathrm{SL}[1\mathrm{G}]$.

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 fork nodes along each path starting from the root (recall that bounded-fork trees are suitable models for $\mathrm{SL}^{\emptyset}[1\mathrm{G}]$). If at most k forks in a path are allowed, the ability to count the number of occurring forks is obtained 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 most i additional forks along a path. Naturally, the initial states belong to Q_k and only states in Q_0, from where no more forks are admitted, can be involved in the Büchi acceptance condition.

▶ **Definition 17** (Bounded-Fork Automaton). *An* NTA $\mathcal{A} \triangleq \langle \Sigma, \Lambda, \mathrm{Q}, \delta, q_I, \mathrm{Q}_\mathrm{F} \rangle$ *is* k-forking *(*k-NTA*, for short), for some given* $k \in \mathbb{N}$*, if* (i) $1 \in \Lambda$ *and* (ii) *there is a* $(k+1)$*-partition* $(\mathrm{Q}_0, \ldots, \mathrm{Q}_k)$ *of* Q *satisfying the following constraints:* (a) $q_I \in \mathrm{Q}_k$*;* (b) $\mathrm{Q}_\mathrm{F} \subseteq \mathrm{Q}_0$*;* (c) *for all indexes* $i \in [0,k]$*, states* $q \in \mathrm{Q}_i$*, input symbols* $\sigma \in \Sigma$*, and node degrees* $d \in \Lambda$*, if* $d = 1$ *then* $\delta(q,\sigma,d) \subseteq \bigcup_{j=0}^{i} \mathrm{Q}_j$ *else* $\delta(q,\sigma,d) \subseteq (\bigcup_{j=0}^{i-1} \mathrm{Q}_j)^d$*.*

The motivation for using k-NTAs, instead of standard NTAs, is clearly expressed by Theorem 18, 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 [42]. To prove the result, we devise a recursive reachability algorithm that looks for a reachable cycle that 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 backtrack 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, as well as to Büchi games, where the number of turns assigned to the universal player is limited by k [39].

▶ **Theorem 18.** *The emptiness problem for a* k-NTA *with* n *states and transition function of size* m *can be solved in* $\textsc{Space}(k \cdot \log n + \log^2 n + \log m)$ *and* $\textsc{ATime}[k\text{-ALT}](\log n + \log m)$*.*

Good for Game 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 read, along all the possible branching directions [36, 40]. This approach works pretty nicely for deterministic word automata, but not for general nondeterministic ones, as the nondeterministic choice at a given instant of time can be solved by exploiting knowledge about future 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 non-determinism. This leads to the notion of nondeterministic *good-for-game automata* [27].

Fixed an *a priori* set of node degrees $\Lambda \subseteq \mathbb{N}_+$, the *word-on-tree function* $\mathsf{wot}_\Lambda \colon \mathrm{NWA} \to \mathrm{NTA}$ maps an NWA $\mathcal{W} = \langle \Sigma, \mathrm{Q}, \delta, q_I, \mathrm{Q}_\mathrm{F} \rangle$ to the NTA $\mathsf{wot}_\Lambda(\mathcal{W}) \triangleq \left\langle \Sigma, \Lambda, \mathrm{Q}, \widehat{\delta}, q_I, \mathrm{Q}_\mathrm{F} \right\rangle$, whose tree transition function $\widehat{\delta}$ is derived from the word transition function δ as follows: $\widehat{\delta}(q,\sigma,d) \triangleq \prod_{i=0}^{d-1} \delta(q,\sigma)$, for all states $q \in \mathrm{Q}$, input symbols $\sigma \in \Sigma$, and node degree $d \in \Lambda$.

▶ **Definition 19** (Good-for-Game Automaton). *Let* $\mathcal{T}$ *be a class of* Σ*-labelled* Λ*-trees. An* NWA $\mathcal{W} = \langle \Sigma, \mathrm{Q}, \delta, q_I, \mathrm{Q}_\mathrm{F} \rangle$ *is a* good-for-game automaton w.r.t. $\mathcal{T}$ *(*$\mathcal{T}$-GFG*, for short) if* $\mathrm{Trc}(\mathcal{T}) \subseteq \mathsf{L}(\mathcal{W})$ *implies* $\mathcal{T} \in \mathsf{L}(\mathsf{wot}_\Lambda(\mathcal{W}))$*, where* $\mathrm{Trc}(\mathcal{T})$ *is the set of* Σ*-traces of* $\mathcal{T}$*, for all trees* $\mathcal{T} \in \mathcal{T}$*.*

Intuitively, good-for-game automata only use knowledge of the (non-strict) past to determine the next steps and no information on the future (via forks related to nondeterministic

guesses). This relates to the notion of strategy in the context of games, where strategies are purely based on histories (*i.e.*, finite traces). In particular, note that the converse direction, $\mathcal{T} \in \mathsf{L}(\mathsf{wot}_\Lambda(\mathcal{W}))$ implies $\mathrm{Trc}(\mathcal{T}) \subseteq \mathsf{L}(\mathcal{W})$, always holds true.

The next paragraph introduces a class of word automata that are GFG for k-bounded trees and will allow us to define a satisfiability algorithm for $\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]$.

Prefix-Deterministic Word Automata. *Prefix-deterministic word automata* are NWAs which behave deterministically on arbitrary prefixes of their runs and behave freely, *i.e.*, nondeterministically, afterwards. The idea is that such automata can be used to encode the checks for compliance of all the paths of a tree *w.r.t.* an LTL property. We shall take advantage of the prefix-determinism to constrain the nondeterministic choices within the automaton to occur only after an initial prefix where all the forks already occurred.

▶ **Definition 20** (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, for all infinite words* $v \cdot w \in \Sigma^\omega$*, it holds that* $v \cdot w \in \mathsf{L}(\mathcal{W})$ iff *either one of the following two conditions holds true, where* $q_v \triangleq \widetilde{\delta}^*(q_I, v)$*: 1)* $q_v = \top$*; 2)* $q_v \neq \bot$ *and* $w \in \mathsf{L}(\mathcal{W}_v)$ *with* $\mathcal{W}_v \triangleq \langle \Sigma, \mathrm{Q}, \delta, q_v, \mathrm{Q_F} \rangle$*.*

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 $\mathcal{W}$ itself to complete the behaviours.

▶ **Theorem 21.** *For every* NWA $\mathcal{W}$ *with* n *states, there exists a* PD-NWA $\mathcal{D}$ *with* $n + 2^n$ *states such that* $\mathsf{L}(\mathcal{D}) = \mathsf{L}(\mathcal{W})$*.*

The standard automaton-theoretic construction for LTL [41] can be easily lifted to PD-NWA as stated by Theorem 22. Notice that, when the WLTL fragment of LTL is considered, the automaton construction has to deal with a number of formulas which is only singly-exponential *w.r.t.* the size of the input formula.

▶ **Theorem 22.** *For every* LTL *(resp.* WLTL*) formula* ψ*, there is a* PD-NWA $\mathcal{D}_\psi$ *with* $2^{\mathrm{O}\left(2^{|\psi|}\right)}$ *(*resp.*,* $\mathrm{O}\left(2^{|\psi|^3}\right)$*) states such that* $\mathsf{L}(\mathcal{D}_\psi) = \mathsf{L}(\psi)$*.*

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

6 The Satisfiability Problem

We solve satisfiability for $\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]$ by reducing it to non-emptiness of a bounded-fork automaton. Let φ be an arbitrary $\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]$ sentence and, thanks to Proposition 8, $\eth \triangleq \langle \zeta, \Phi, \ell \rangle$ be its a corresponding Skolem skeleton. Corollary 16 tells us that φ is satisfiable *iff* $\eth$ is single-time normally satisfied by a k-fork CGS, with $k \triangleq \mathsf{max}\{0, |\Phi| - 1\}$. We construct a k-NTA $\mathcal{N}_\varphi$ recognising all normal models of $\eth$ (which are 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 deterministic safety automaton checking that the structure of the input tree complies with Definition 14; *(2)* the nondeterministic Büchi automaton $\mathcal{N}_\Phi$ ensures 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 bindings 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 Item b of Definition 14 on the global binding function g. In particular, the local coherence condition precisely corresponds to the property required on the local binding function l by Item b of the same definition. Obviously, the branching degree of the tree 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 $\widehat{\mathsf{r}}\colon \mathrm{X} \in 2^{\mathrm{B}} \mapsto ([0, |\mathsf{muc}(\mathrm{X})| - 1] \to \mathsf{muc}(\mathrm{X}))$ such that, for each set of bindings $\mathrm{X} \subseteq \mathrm{B}$, the associated map $\widehat{\mathsf{f}} \triangleq \widehat{\mathsf{r}}(\mathrm{X})\colon [0, |\mathsf{muc}(\mathrm{X})| - 1] \to \mathsf{muc}(\mathrm{X})$ is bijective. This function is used in the following construction to ensure Item a of Definition 14.

▶ **Construction 1** (Structure Automaton). *The* structure automaton $\mathcal{D}_\eth$ *is the safety* DTA $\langle \Sigma, \Lambda, 2^{\mathrm{B}}, \delta, \emptyset \rangle$ *whose transition function* δ *is defined as follows:* $\delta(\mathrm{X}, \sigma, d) \triangleq \prod_{i=0}^{d-1} \widehat{\mathsf{f}}(i)$*, where* $\widehat{\mathsf{f}} \triangleq \widehat{\mathsf{r}}(\sigma \cap \mathrm{B})$*, if* $\mathrm{X} \subseteq \sigma$ *and* $d = |\mathsf{img}\left(\widehat{\mathsf{f}}\right)|$*, and* $\delta(\mathrm{X}, \sigma, d) \triangleq \bot$*, otherwise, for any set of bindings* $\mathrm{X} \subseteq \mathrm{B}$*, input symbol* $\sigma \in \Sigma$*, and node degree* $d \in \Lambda$*.*

It can be shown that, if we extend any normal model of $\eth$ with the binding labelling dictated by its global binding function, we obtain a tree structure that is accepted by $\mathcal{D}_\eth$. *Vice versa*, every tree accepted by $\mathcal{D}_\eth$ is the backbone of a tree CGS, whose functions l, g, and r (see Definition 14) can be easily 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 22, for any $\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]$ (*resp.*, $\mathrm{WSL}^{\not\emptyset}[1\mathrm{G}]$) 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 using a constant number of additional states, we can turn $\mathcal{D}_\psi$ into a PD-NWA $\mathcal{D}_{\widetilde{\phi}}$ recognising all models of the LTL (*resp.*, WLTL) formula $\widetilde{\phi} \triangleq \mathtt{G}\ (\ell(\phi) \to ((\mathtt{X}\,\mathtt{G}\ \neg\ell(\phi)) \wedge (\psi \vee \mathtt{F}\ \neg\flat)))$. 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* NTA*s* $\mathsf{wot}_\Lambda(\mathcal{D}_{\widetilde{\phi}})$ *derived from the* PD-NWA $\mathcal{D}_{\widetilde{\phi}}$*, for each* $\phi \in \Phi$*.*

While neither $\mathcal{D}_\eth$ nor $\mathcal{N}_\Phi$ taken in isolation is a bounded-fork automaton, their product does satisfy the property. Indeed, $\mathcal{N}_\Phi$ ensures that, along a path, the labelling $\ell(\phi)$ of a sentence $\phi \in \Phi$ occurs at most once and $\mathcal{D}_\eth$ has a transition with more than one successor at a given node v only if the set of bindings in the labelling of v does not unify. In the remaining part of this section, by $\mathrm{SL}^{\not\emptyset}_k[1\mathrm{G}]$ (*resp.*, $\mathrm{WSL}^{\not\emptyset}_k[1\mathrm{G}]$) we denote the set of those $\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]$ (*resp.*, $\mathrm{WSL}^{\not\emptyset}[1\mathrm{G}]$) sentences containing at most k subsentences.

▶ **Theorem 24.** *The* NTA $\mathcal{N}_\varphi$ *is* k*-forking, for every* $\mathrm{SL}^{\not\emptyset}_k[1\mathrm{G}]$ *sentence* φ*.*

At this point, it is clear that every tree accepted by $\mathcal{N}_\varphi$ corresponds to a normal model for the Skolem skeleton $\eth$ of φ. *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 a GFG for the class of bounded-fork tree, as shown in Theorem 23. This leads to the following result.

▶ **Theorem 25.** *For every* $\mathrm{SL}_k^{\not\emptyset}[1\mathrm{G}]$ *(*resp.*,* $\mathrm{WSL}_k^{\not\emptyset}[1\mathrm{G}]$*) 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 *itself.*

By Theorem 18, we obtain that deciding $\mathrm{WSL}^{\not\emptyset}[1\mathrm{G}]$ is provably easier than deciding full SL[1G], which is known to be 2ExpTime-complete, while the complexity for $\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]$ is lower only if we assume the widely-shared conjecture that ExpSpace $\subset$ 2ExpTime. While PSpace-hardness of $\mathrm{WSL}^{\not\emptyset}[1\mathrm{G}]$ trivially follows from an obvious encoding of standard modal-logic satisfiability, it is not even known whether $\mathrm{SL}^{\not\emptyset}[1\mathrm{G}]$ is ExpTime-hard.

▶ **Theorem 26.** $\mathrm{SL}_k^{\not\emptyset}[1\mathrm{G}]$ *(*resp.*,* $\mathrm{WSL}^{\not\emptyset}[1\mathrm{G}]$*) satisfiability problem is* AExpTime[k-alt] *(*resp.*,* PSpace-complete*).*

7 Discussion

We have considered efficiently decidable fragments of one-goal SL, 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 is the number of subsentences within the formulae. If, however, those arguments are further restricted to Boolean formulae, a PSpace-complete result is given. Both the 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 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 admit a bounded number of branching points along any computation. Second, we leverage a novel class of automata, called bounded-fork tree automata, that can recognise normal models and whose language emptiness problem can be checked in LogSpace.

Since SL[1G] strictly includes both ATL* and CTL*, the results immediately applies 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 complexity, which is PSpace-complete, is strictly lower than the one for the full languages, known to be 2ExpTime-complete.

Acknowledgments

Partially supported by the GNCS 2020 project “Ragionamento Strategico e Sintesi Automatica di Sistemi Multi-Agente”.

References

1 E. Acar, M. Benerecetti, and F. Mogavero. Satisfiability in Strategy Logic Can Be Easier than Model Checking. In *AAAI Press19*, pages 2638–2645. AAAI Press, 2019.

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

3 E.G. Amparore, S. Donatelli, and F. Gallà. A CTL* Model Checker for Petri Nets. In *PETRI NETS'20*, LNCS 12152, pages 403–413. Springer, 2020.

4 F. Baader and W. Snyder. Unification Theory. In *Handbook of Automated Reasoning (vol. 1).*, pages 445–532. Elsevier & MIT Press, 2001.

5 M. Benerecetti, F. Mogavero, and A. Murano. Substructure Temporal Logic. In *LICS'13*, pages 368–377. IEEECS, 2013.

6 M. Benerecetti, F. Mogavero, and A. Murano. Reasoning About Substructures and Games. *TOCL*, 16(3):25:1–46, 2015.

7 P. Bouyer, P. Gardy, and N. Markey. Weighted Strategy Logic with Boolean Goals Over One-Counter Games. In *FSTTCS'15*, LIPIcs 45, pages 69–83. Leibniz-Zentrum fuer Informatik, 2015.

8 P. Bouyer, P. Gardy, and N. Markey. On the semantics of Strategy Logic. *IPL*, 116(2):75–79, 2016.

9 S. Bova and F. Mogavero. Herbrand Property, Finite Quasi-Herbrand Models, and a Chandra-Merlin Theorem for Quantified Conjunctive Queries. In *LICS'17*, pages 1–12. ACM, 2017.

10 K. Chatterjee, T.A. Henzinger, and N. Piterman. Strategy Logic. In *CONCUR'07*, LNCS 4703, pages 59–73. Springer, 2007.

11 K. Chatterjee, T.A. Henzinger, and N. Piterman. Strategy Logic. *IC*, 208(6):677–693, 2010.

12 H. Comon and V. Cortier. Flatness Is Not a Weakness. In *CSL'00*, LNCS 1862, pages 262–276. Springer, 2000.

13 D. Dams. Flat Fragments of CTL and CTL*: Separating the Expressive and Distinguishing Powers. *LJIGPL*, 7(1):55–78, 1999.

14 S. Demri, R. Lazic, and D. Nowak. On the Freeze Quantifier in Constraint LTL: Decidability and Complexity. *IC*, 205(1):2–24, 2007.

15 S. Demri and P. Schnoebelen. The Complexity of Propositional Linear Temporal Logics in Simple Cases. *IC*, 174(1):84–103, 2002.

16 E.A. Emerson. Temporal and Modal Logic. In *Handbook of Theoretical Computer Science (vol. B).*, pages 995–1072. MIT Press, 1990.

17 E.A. Emerson and J.Y. Halpern. "Sometimes" and "Not Never" Revisited: On Branching Versus Linear Time. In *POPL'83*, pages 127–140. ACM, 1983.

18 E.A. Emerson and J.Y. Halpern. "Sometimes" and "Not Never" Revisited: On Branching Versus Linear Time. *JACM*, 33(1):151–178, 1986.

19 G.E. Fainekos, S.G. Loizou, and G.J. Pappas. Translating Temporal Logic to Controller Specifications. In *DC'06*, pages 899–904. IEEECS, 2006.

20 P. Gardy, P. Bouyer, and N. Markey. Dependences in Strategy Logic. In *STACS'18*, LIPIcs 96, pages 34:1–15. Leibniz-Zentrum fuer Informatik, 2018.

21 P. Gardy, P. Bouyer, and N. Markey. Dependences in Strategy Logic. *TCS*, 64(3):467–507, 2020.

22 S. Ghosh and R. Ramanujam. Strategies in Games: A Logic-Automata Study. In *ESSLLI'11*, LNCS 7388, pages 110–159. Springer, 2011.

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

24 J. Gutierrez, P. Harrenstein, and M. Wooldridge. Iterated Boolean Games. In *IJCAI'13*, pages 932–938, 2013.

25 J. Gutierrez, P. Harrenstein, and M. Wooldridge. Reasoning about Equilibria in Game-Like Concurrent Systems. In *KR'14*, pages 408–417. AAAI Press, 2014.

26 J. Gutierrez, P. Harrenstein, and M. Wooldridge. Iterated Boolean Games. *IC*, 242:53–79, 2015.

27 T.A. Henzinger and N. Piterman. Solving Games Without Determinization. In *CSL'06*, LNCS 4207, pages 395–410. Springer, 2006.

28 O.H. Ibarra and Z. Dang. On Removing the Pushdown Stack in Reachability Constructions. In *ISAAC'01*, LNCS 2223, pages 244–256. Springer, 2001.

29 B. Khoussainov and A. Nerode. *Automata Theory and Its Applications.* Birkhauser, 2001.

30 O. Kupferman, M.Y. Vardi, and P. Wolper. An Automata Theoretic Approach to Branching-Time Model Checking. *JACM*, 47(2):312–360, 2000.

31 F. Mogavero, A. Murano, G. Perelli, and M.Y. Vardi. What Makes ATL* Decidable? A Decidable Fragment of Strategy Logic. In *CONCUR'12*, LNCS 7454, pages 193–208. Springer, 2012.

32 F. Mogavero, A. Murano, G. Perelli, and M.Y. Vardi. Reasoning About Strategies: On the Model-Checking Problem. *TOCL*, 15(4):34:1–42, 2014.

33 F. Mogavero, A. Murano, G. Perelli, and M.Y. Vardi. Reasoning About Strategies: On the Satisfiability Problem. *LMCS*, 13(1:9):1–37, 2017.

34 F. Mogavero, A. Murano, and M.Y. Vardi. Reasoning About Strategies. In *FSTTCS'10*, LIPIcs 8, pages 133–144. Leibniz-Zentrum fuer Informatik, 2010.

35 F. Mogavero and G. Perelli. Binding Forms in First-Order Logic. In *CSL'15*, LIPIcs 41, pages 648–665. Leibniz-Zentrum fuer Informatik, 2015.

36 A. Pnueli and R. Rosner. On the Synthesis of a Reactive Module. In *POPL'89*, pages 179–190. ACM, 1989.

37 S. Schewe. ATL* Satisfiability is 2ExpTime-Complete. In *ICALP'08*, LNCS 5126, pages 373–385. Springer, 2008.

38 P. Schnoebelen. The Complexity of Temporal Logic Model Checking. In *AIML'02*, pages 393–436. College Publications, 2002.

39 L.J. Stockmeyer. The Polynomial-Time Hierarchy. *TCS*, 3(1):1–22, 1976.

40 M.Y. Vardi. Reasoning about The Past with Two-Way Automata. In *ICALP'98*, LNCS 1443, pages 628–641. Springer, 1998.

41 M.Y. Vardi and P. Wolper. An Automata-Theoretic Approach to Automatic Program Verification. In *LICS'86*, pages 332–344. IEEECS, 1986.

42 M.Y. Vardi and P. Wolper. Automata-Theoretic Techniques for Modal Logics of Programs. *JCSS*, 32(2):183–221, 1986.