The Complexity of Regularity in Grammar Logics (long version)

Stéphane Demri *

Lab. Spécification et Vérification
ENS de Cachan & CNRS UMR 8643
61 Av. Pdt. Wilson
94235 Cachan Cedex, France
email: demri@lsv.ens-cachan.fr

March 29, 2001

Abstract

A modal reduction principle of the form $[i_1]\ldots[i_n]\mathrm{p} \Rightarrow [j_1]\ldots[j_{n'}]\mathrm{p}$ can be viewed as a production rule $i_1 \cdot \ldots \cdot i_n \rightarrow j_1 \cdot \ldots \cdot j_{n'}$ in a formal grammar. We study the extensions of the multimodal logic K_m with m independent K modal connectives by finite addition of axiom schemes of the above form such that the associated finite set of production rules forms a regular grammar. We show that given a regular grammar $\mathcal{G}$ and a modal formula ϕ, deciding whether the formula is satisfiable in the extension of K_m with axiom schemes from $\mathcal{G}$ can be done in deterministic exponential-time in the size of $\mathcal{G}$ and ϕ, and this problem is complete for this complexity class. Such an extension of K_m is called a regular grammar logic. The proof of the exponential-time upper bound is extended to PDL-like extensions of K_m and to global logical consequence and global satisfiability problems. Using an equational characterization of context-free languages, we show that by replacing the regular grammars by linear ones, the above problem becomes undecidable. The last part of the paper presents non-trivial classes of exponential time complete regular grammar logics.

*On leave from Laboratoire LEIBNIZ, Grenoble, France.

This is a long version of a paper to appear in Journal of Logic and Computation.

Keywords: computational complexity, modal logic, formal grammar, regular language, finite automaton.

1 Introduction

Capturing the decidability/complexity status of modal logics. A nowadays popular approach to establish the decidability of modal logics consists in studying the decidability status of fragments of first-order logic [Var97, ANB98] (see also [Gab81]). Sometime, these fragments are augmented by features that are not present in the standard first-order language allowing more expressive power, often at the cost of losing decidability (see e.g. [GOR97]). By contrast, the guarded fixed point logic μLGF [GW99] is a decidable fragment of fixed-point first-order logic in which can be naturally embedded the modal μ-calculus (see e.g. [Koz83]). Once an interesting decidable fragment is identified, the design of decision procedures that meet the best worst-case complexity upper bounds is often the next step (see e.g. [Sch97, Niv98, Hus99]). However, even if your favorite modal logic can be embedded in a known decidable fragment of second-order logic, the characterization of the worst-case complexity of your logic is not straightforward from the complexity of the second-order fragment. For instance, the standard modal logic K can be embedded into FO2, the fragment of classical predicate logic with only two individual variables and without function symbols but FO2-satisfiability is **NEXPTIME**-complete whereas K-satisfiability is "only" **PSPACE**-complete [Lad77] (see e.g. [Pap94] for a thorough introduction to computational complexity). Therefore, there is a need to develop general methods dedicated to the computational complexity of modal logics. Spaan's thesis [Spa93] can be considered as an important step towards this direction. Indeed, the worst-case complexity of independent fusion of modal logics is studied there (see also [Hem94]). The study of the complexity of PDL-like logics (e.g. the modal μ-calculus, Combinatory PDL) can be understood as the modal counterpart for the study of decidable fragments of second-order logic. Indeed, many modal logics can be naturally embedded into PDL-like logics (see e.g. [Tuo90, Sch91, Gia95]) and therefore the design of efficient decision procedures for PDL-like logics is another way to study the complexity of modal logics in a uniform frame-

work. Typically, there is some natural transformation from satisfiability for the standard modal logics B, S4, S5 into PDL with converse (see e.g. [FL79]). However, such translations have not been studied in a systematic way. In the present paper, the main object of study is a countably infinite class of polymodal logics such that the satisfiability problem can be embedded in linear-time to first-order logic. Unfortunately, the target fragment is not known to belong to identified decidable fragments and therefore decidability and complexity shall be established by modal-like techniques partly based on formal language theory.

Grammar logics. An important idea in logic programming is to translate the special purpose formalism of formal (context-free) grammars into a general purpose one, namely first-order predicate logics. In [FdCP88], a similar approach is suggested where the analysis or generation of a sentence is transformed to theorem proving for modal logics. The modal logics (called "grammar logics") introduced in [FdCP88] are closely related to formal grammars. Namely, with each production rule $i_1 \cdot \ldots \cdot i_n \rightarrow j_1 \cdot \ldots \cdot j_{n'}$ in the grammar is associated a modal axiom $[i_1] \ldots [i_n]\mathrm{p} \Rightarrow [j_1] \ldots [j_{n'}]\mathrm{p}$. Such axioms are called *reduction principles* in [Ben76, CS94] and they are a special type of Sahlqvist formulae [Sah75] and primitive formulae [Kra96]. They are typical in modal logic. In this paper, we study the extensions of the multimodal logic K_m with m independent K modal connectives by finite addition of axiom schemes of the above form such that the associated finite set of production rules forms a regular formal grammar.

Having in mind the initial motivation to introduce the grammar logics in [FdCP88], it is worth observing that the regular grammar logics are too expressive to encode the generation of strings by regular grammars. Indeed, whether a string belongs to a context-free language (defined by a context-free grammar) is a **P**-complete problem (see e.g. [JL76, Corollary 11]) whereas the satisfiability problem of any regular grammar logic shall be shown to be **PSPACE**-hard. In a sense, introducing regular grammar logics to analyze regular languages is not very efficient. However, we claim that it is more interesting to take advantage of the wealth of knowledge about regular languages in order to analyze the computational complexity of regular grammar logics.

Related modal logics. Although the grammar logics may seem artificial, many polymodal logics containing fragments that are regular

grammar logics can be found in the literature (see e.g. [FL79, Cat89, HM92, Gas94, FdCH95, Hem96, HM97] to quote a few examples). More importantly, Description Logics (DLs) that are used to represent terminological knowledge (see e.g. [SSS91]) are strongly related to grammar logics. A current line of research in DLs community consists in studying more and more expressive description logics as soon as they are meaningful for knowledge representation languages (see e.g. [Wol99, HST00]). Typically, in order to obtain expressive roles, one can either add role constructors or constraint the interpretation of roles. Inverse roles, transitive roles, role value inclusions and role hierarchies are features of DLs that allow a gain of expressive power for roles (see e.g. [HS99, HM00, HST00, Are00]). The grammatical constraints on relations considered in the present paper can be viewed as role inclusion axioms for roles built from atomic roles and finite composition.

That is why understanding the complexity of grammar logics can help understanding the complexity of other related modal logics, including DLs.

Our contribution. We show that every regular grammar logic is decidable. Up to now, it is only known that every right linear grammar logic is decidable [Bal98, BGM98] and this solves an open problem mentioned in [Bal98, BGM98]. It is worth observing that although the right linear grammars generate the same class of languages as the left linear grammars, this correspondence is not relevant at the level of regular grammar logics. It is fair to mention that the initial motivation for this work was to understand why the decidability proof in [Bal98, BGM98] cannot be naturally extended to left linear grammars. Our decidability proof consists in defining transformations into the satisfiability problem for PDL. This allows us to prove that the satisfiability problem for any regular grammar logic is in **EXPTIME** and this result can be extended to PDL-like logics, to description logics with inclusion axioms (some restrictions are made here), to the global logical consequence problem and to the global satisfiability problem. Up to now, it is only known that the satisfiability problem for any right linear grammar logics is in **NEXPTIME** [Bal98, BGM98]. Unfortunately, our transformation is not in polynomial-time in the size of the regular grammars. We then show that given a regular grammar $\mathcal{G}$ and a modal formula ϕ, deciding whether the formula is satisfiable in the extension of K_m with axiom schemes from $\mathcal{G}$ can be done in deterministic exponential-time in the size of $\mathcal{G}$ and ϕ.

We refer to this problem as the *general satisfiability problem* for regular grammar logics. The complexity upper bound is established by defining a polynomial time transformations into the satisfiability problem for PDL with finite automata. PDL with automata is a succinct variant of PDL where the regular expressions are replaced by finite automata. Up to now, it is only known that the general satisfiability problem restricted to right linear grammars is in **N2EXPTIME** [Bal98, BGM98]. In [Bal98, BGM98], it is shown that the general satisfiability problem for context-free grammars logics is undecidable by reducing the empty intersection problem for context-free grammars which is undecidable (see e.g. [RS94]). The proof is based on the completeness of tableaux-based proof systems. Alternatively, we prove here that the general satisfiability problem for linear grammar logics is undecidable and the core of our proof uses a method based on properties of formal languages only. The undecidability proof is extended to other classes of context-free grammar logics.

Finally, we present non-trivial classes of **EXPTIME**-complete grammar logics and a class of **PSPACE**-complete regular grammar logics. Throughout the paper, we illustrate the main complexity results by examples. Moreover, none of the proofs are based on tableaux techniques as in [Bal98, BGM98] (see e.g. the proofs of Theorem 6 and Lemma 27).

Related work. Formal language theory and automata theory are already used for logics such as modal μ-calculus, Propositional Dynamic Logic PDL, Propositional Temporal Logic PTL, CTL* (see e.g. [VW86, VW94, EJ99]) but from a different perspective than here. We can also mention the Extended Temporal Logic (ETL) that can express properties of a sequence definable by a right linear grammar [Wol83, VW94]. In our work, we are only dealing with automata on finite words. Furthermore, our work continues the line of research relating regular expressions and Propositional Dynamic Logic (see e.g. [BM75, Pra79, FL79, Pra81, HPS83, HKT00]).

Plan of the paper. The rest of the paper is structured as follows. In Section 2 and Section 3, we define the class of grammar logics as well as other PDL-like logics. Some basic definitions about formal grammars are also recalled. In Section 4, we show that the general satisfiability problem for regular grammar logics is **EXPTIME**-complete. In Section 5 we extend this result to other logical problems. In Section

6, we show that the general satisfiability problem for linear grammar logics is undecidable by using an equational characterization of context-free languages. In Section 7, we show that every regular grammar logic has a **PSPACE**-hard satisfiability problem and we introduce a class of **PSPACE**-complete regular grammar logics. In Section 8, we present various classes of **EXPTIME**-complete regular grammar logics. Section 9 concludes the paper by mentioning open problems.

2 Logics

A *modal language* L is determined by a countable set Π of *modal expressions* and by a countable set PRP of atomic formulae. Modal expressions in Π are intended to represent (indices of) binary relations that determine modal connectives. The set FOR of L-formulae is defined as the smallest set such that $\mathrm{PRP} \subseteq \mathrm{FOR}$ and, if $\phi, \psi \in \mathrm{FOR}$, then $\phi \wedge \psi \in \mathrm{FOR}$, $\neg\phi \in \mathrm{FOR}$ and for $\pi \in \Pi$, $[\pi]\phi \in \mathrm{FOR}$. Standard abbreviations include $\Rightarrow$, $\langle\pi\rangle$, $\top$.

Example 1 For $m \geq 1$, we write L_m to denote the modal language determined by the set $\{\mathtt{c}_i : i \in \{1, \ldots, m\}\}$ of modal expressions (say "program constants") and by the set $\mathrm{PRP} = \{\mathrm{p}_i : i \geq 0\}$ of propositional variables. By abusing our notations, we often identify $\{\mathtt{c}_i : i \in \{1, \ldots, m\}\}$ with $\{1, \ldots, m\}$. We write $\mathrm{L}_{\mathbb{N}}$ to denote the modal language determined by the set $\{\mathtt{c}_i : i \geq 1\}$ of modal expressions and by the set $\mathrm{PRP} = \{\mathrm{p}_i : i \geq 0\}$ of propositional variables.

Given a set $\Pi_0 = \{\mathtt{c}_i : i \geq 1\}$ of *program constants*, a set $\{\mathrm{p}_i : i \geq 0\}$ of propositional variables and a set $\{\mathtt{i}_i : i \geq 0\}$ of nominals (interpreted by singletons in the possible-worlds semantics), the set Π_{full} of modal expressions and the set FOR_{full} of formulae for the language L_{full} are inductively defined as follows using the BNF notation:

$$\Pi_{full} \ni \pi ::= \mathtt{c}_i \mid \mathrm{U} \mid \pi \cup \pi' \mid \pi;\pi' \mid \pi^{-1} \mid \pi^* \mid \phi?$$
$$\mathrm{FOR}_{full} \ni \phi ::= \mathrm{p}_i \mid \mathtt{i}_i \mid \phi \wedge \psi \mid \neg\phi \mid [\pi]\phi.$$

In the rest of the paper, we shall study logics whose languages are fragments of L_{full}: the whole language contains all that we need. We write π^α as an abbreviation for $\overbrace{\pi;\ldots;\pi}^{\alpha \text{ times}}$ and by convention π^0 is $\top?$, also noted *id* (also known as *self* in the DL literature).

Given a modal language L, an L-*frame* is a structure $\mathcal{F} = \langle W, (R_\pi)_{\pi \in \Pi} \rangle$ such that W is a nonempty set and for $\pi \in \Pi$, R_π is a binary relation on W. An L-*model* is a structure $\mathcal{M} = \langle W, (R_\pi)_{\pi \in \Pi}, V \rangle$ such that $\langle W, (R_\pi)_{\pi \in \Pi} \rangle$ is an L-frame and V is a valuation $V : \mathrm{PRP} \to \mathcal{P}(W)$. The diagonal relation $\{\langle x, x \rangle : x \in W\}$ is denoted by Id_W. We say that the formula ϕ is satisfied in the model $\mathcal{M}$ by the state x (written $\mathcal{M}, x \models \phi$) if the following conditions are satisfied:

- $\mathcal{M}, x \models \mathrm{p} \stackrel{\text{def}}{\Leftrightarrow} x \in V(\mathrm{p})$ for $\mathrm{p} \in \mathrm{PRP}$;
- $\mathcal{M}, x \models \phi_1 \wedge \phi_2 \stackrel{\text{def}}{\Leftrightarrow} \mathcal{M}, x \models \phi_1$ and $\mathcal{M}, x \models \phi_2$;
- $\mathcal{M}, x \models \neg\phi \stackrel{\text{def}}{\Leftrightarrow}$ not $\mathcal{M}, x \models \phi$;
- $\mathcal{M}, x \models [\pi]\phi \stackrel{\text{def}}{\Leftrightarrow}$ for all $x' \in R_\pi(x)$, $\mathcal{M}, x' \models \phi$ where $R_\pi(x) \stackrel{\text{def}}{=} \{x' \in W : xR_\pi x'\}$.

An L-formula ϕ is said to be *true* in the L-model $\mathcal{M}$ (written $\mathcal{M} \models \phi$) $\stackrel{\text{def}}{\Leftrightarrow}$ for all $x \in W$, $\mathcal{M}, x \models \phi$.

A *modal logic* $\mathcal{L}$ is a pair $\langle \mathrm{L}, \mathcal{S} \rangle$ where L is a modal language and $\mathcal{S}$ is a nonempty class of L-models. The class $\mathcal{S}$ is usually defined in terms of properties that the relations in the models of $\mathcal{S}$ are supposed to satisfy. The set of formulae of the logic $\mathcal{L}$ is naturally defined as the set of L-formulae. An L-formula is said to be $\mathcal{L}$-*satisfiable* $\stackrel{\text{def}}{\Leftrightarrow}$ there is an $\mathcal{L}$-model $\mathcal{M} \in \mathcal{S}$ and $x \in W$ such that $\mathcal{M}, x \models \phi$. An L-formula is said to be $\mathcal{L}$-*valid* $\stackrel{\text{def}}{\Leftrightarrow}$ for all the $\mathcal{L}$-models $\mathcal{M} \in \mathcal{S}$, $\mathcal{M} \models \phi$.

Example 2 The *standard* L_{full}-*models* are the L_{full}-models of the form $\mathcal{M} = \langle W, (R_\pi)_{\pi \in \Pi_{full}}, V \rangle$ such that for $i \geq 0$, $\pi, \pi' \in \Pi_{full}$, $\phi \in \mathrm{FOR}_{full}$ we have:

- $V(\mathtt{i}_i)$ is a singleton set $\quad R_{\phi?} = \{\langle x, x \rangle : \mathcal{M}, x \models \phi\}$; $R_{\mathrm{U}} = W \times W$;
- $R_{\pi \cup \pi'} = R_\pi \cup R_{\pi'} \quad R_{\pi;\pi'} = R_\pi \circ R_{\pi'} \quad R_{\pi^*} = R_\pi^* \quad R_{\pi^{-1}} = R_\pi^{-1}$.

Many PDL-like logics can be then defined by restricting the language L_{full} and by considering the natural corresponding restriction on the class of standard L_{full}-models.

- Combinatory Propositional Dynamic Logic with Converse (converse-CPDL) [PT91]: based on L_{full};
- Combinatory Propositional Dynamic Logic (CPDL) [PT91]: based on L_{full} without $^{-1}$;

- Propositional Dynamic Logic (PDL) [FL79, Pra79]: based on L_{full} without $^{-1}$, U and $\{\mathtt{i}_i : i \geq 0\}$;
- Test-free Propositional Dynamic Logic ($\mathrm{PDL}^{(0)}$) [Har84]: based on L_{full} without $^{-1}$, ?, U and $\{\mathtt{i}_i : i \geq 0\}$;
- Test-free Propositional Dynamic Logic with Identity ($\mathrm{PDL}^{(0)+id}$): based on L_{full} without $^{-1}$, ?, U and $\{\mathtt{i}_i : i \geq 0\}$ but with id.

The satisfiability problem of any above logic between $\mathrm{PDL}^{(0)}$ and converse-CPDL is **EXPTIME**-complete (see e.g. [FL79, Pra79, PT91, ABM00]).

For $m \geq 1$, the multimodal logic $\mathcal{L}_m$ is defined as the pair $\langle \mathrm{L}_m, \mathcal{S}_m \rangle$ such that $\mathcal{S}_m$ is the set of all the L_m-models. Similarly, the multimodal logic $\mathcal{L}_{\mathbb{N}}$ is defined as the pair $\langle \mathrm{L}_{\mathbb{N}}, \mathcal{S}_{\mathbb{N}} \rangle$ such that $\mathcal{S}_{\mathbb{N}}$ is the set of all the $\mathrm{L}_{\mathbb{N}}$-models.

3 Grammar logics

In this section, we define the notion of grammar logics. Before its introduction, we recall a few definitions and results about formal languages and grammars.

3.1 Formal languages

For any alphabet Σ (finite set of symbols), we write Σ^* [resp. Σ^+] to denote the set of [resp. non-empty] finite strings built over elements of Σ. As is usual, ϵ denotes the empty string and $u_1 \cdot u_2$ denotes the concatenation of two strings. For any finite string $u \in \Sigma^*$, we write $|u|$ to denote its length and we write u^k to denote the string composed of k copies of u. By convention, $u^0 = \epsilon$. A *language* is defined as a subset of Σ^* for some finite alphabet Σ. We use the following standard operations on languages: union ($\cup$), concatenation ($\cdot$) and iteration (*). For instance, given a language L on Σ, $\mathrm{L}^* \stackrel{\text{def}}{=} \{\epsilon\} \cup \{u_1 \cdot \ldots \cdot u_n : u_1, \ldots, u_n \in \mathrm{L}, n \geq 1\}$.

3.2 Formal grammars

A *(formal) grammar* $\mathcal{G}$ is a quadruple $\mathcal{G} = \langle N, \Sigma, P, S \rangle$ such that N and Σ are disjoint finite sets of *nonterminal symbols* and *terminal symbols*, respectively. P is a finite set of *production rules*, each production rule is of the form $u \to v$ such that $u \in (N \cup \Sigma)^* \cdot N \cdot (N \cup \Sigma)^*$ and $v \in$

$(N \cup \Sigma)^*$. Finally, $S \in N$ is a special symbol called the *start symbol* (see e.g. [HU79]). For the grammar $\mathcal{G}$, the size of $\mathcal{G}$, denoted $|\mathcal{G}|$, is

$$|\mathcal{G}| \stackrel{\text{def}}{=} (card(N)+card(\Sigma)+\Sigma_{u\to v\in P}(|u\cdot v|+1))\times log(card(N)+card(\Sigma)).$$

Let $\Rightarrow_{\mathcal{G}}$ be the direct derivation relation defined as the subset of $(N \cup \Sigma)^* \times (N \cup \Sigma)^*$ such that $u \Rightarrow_{\mathcal{G}} v \stackrel{\text{def}}{\Leftrightarrow}$ there is a production rule $u' \to v' \in P$ such that $u = u_1 \cdot u' \cdot u_2$, $v = u_1 \cdot v' \cdot u_2$, $u_1, u_2 \in (N \cup \Sigma)^*$. Let $\Rightarrow^*_{\mathcal{G}}$ be the reflexive and transitive closure of $\Rightarrow_{\mathcal{G}}$. The language generated by $\mathcal{G}$, denoted $\mathrm{L}(\mathcal{G})$ is the set of strings $\{u \in \Sigma^* : S \Rightarrow^*_{\mathcal{G}} u\}$. For $i \in N \cup \Sigma$, we write $\mathrm{L}_i(\mathcal{G})$ to denote the set of terminal strings $\{u \in \Sigma^* : i \Rightarrow^*_{\mathcal{G}} u\}$ generated by the symbol i. For instance, for $i \in \Sigma$, $\mathrm{L}_i(\mathcal{G}) = \{i\}$.

The well-known Chomsky hierarchy of grammars and languages is defined by imposing conditions on the form of the production rules of the grammars. A grammar $\mathcal{G}$ is

- *context-free* (in CF) $\stackrel{\text{def}}{\Leftrightarrow}$ all the production rules $u \to v$ satisfy $u \in N$;
- *linear* (in LIN) $\stackrel{\text{def}}{\Leftrightarrow}$ all the production rules $u \to v$ satisfy $u \in N$ and at most one nonterminal occurs in v;
- *right linear* (in RLIN) $\stackrel{\text{def}}{\Leftrightarrow}$ $\mathcal{G}$ is linear and all the production rules $u \to v$ satisfy $v \in \Sigma^* \cup \Sigma^* \cdot N$;
- *left linear* (in LLIN) $\stackrel{\text{def}}{\Leftrightarrow}$ $\mathcal{G}$ is linear and all the production rules $u \to v$ satisfy $v \in \Sigma^* \cup N \cdot \Sigma^*$.

It is known that a language is generated by a right linear grammar iff it is generated by a left linear grammar. A language is said to be *regular* iff there is a right linear grammar that generates it. By extension, by a regular grammar (in REG) we mean either a left linear grammar or a right linear grammar. This is slightly different from the usual convention in the formal language theory literature but it simplifies subsequent developments.

Example 3 Let $\mathcal{G}$ be the left linear grammar $\mathcal{G} = \langle\{1\}, \{2\}, \{1 \to 1 \cdot 2, 1 \to \epsilon\}, 1\rangle$. We have $\mathrm{L}(\mathcal{G}) = \mathrm{L}_1(\mathcal{G}) = \{2^\alpha : \alpha \geq 0\} = \{2\}^*$.

In the technical developments of the paper, unless otherwise stated we assume that each grammar $\mathcal{G} = \langle N, \Sigma, P, S\rangle$ satisfies $N = \{1, \ldots, k\}$ for some $k \geq 1$, $\Sigma = \{k+1, \ldots, m\}$ for some $k \leq m$ (we allow Σ to be empty) and $S = 1$.

3.3 A known characterization of regular languages

We recall below standard facts about regular formal languages. They are included to make the paper more self-contained. An introduction to regular languages can be found in [Per90]. Regular languages coincide with languages accepted by finite automata. Regular languages also coincide with languages generated by *regular expressions*. Let Σ be a finite alphabet. The set $\mathcal{E}_\Sigma$ of regular expressions on the alphabet Σ are the expressions below using the BNF notation: $e ::= \emptyset \mid \epsilon \mid i \mid e_1 \cup e_2 \mid e_1 \cdot e_2 \mid e^*$ where $i \in \Sigma$. The language $\mathrm{LAN}(e)$ represented by the regular expression e is inductively defined as follows:

- $\mathrm{LAN}(\emptyset) = \emptyset \quad \mathrm{LAN}(\epsilon) = \{\epsilon\} \quad \mathrm{LAN}(i) = \{i\}$;
- $\mathrm{LAN}(e_1 \cup e_2) = \mathrm{LAN}(e_1) \cup \mathrm{LAN}(e_2)$;
- $\mathrm{LAN}(e_1 \cdot e_2) = \mathrm{LAN}(e_1) \cdot \mathrm{LAN}(e_2) \quad \mathrm{LAN}(e^*) = \mathrm{LAN}(e)^*$.

The symbols $\emptyset$, ϵ, $\cup$, $\cdot$ and * are overloaded here. We will reduce satisfiability for regular grammar logics into $\mathrm{PDL}^{(0)+id}$-satisfiability. The regular expressions are the terms that allow us to relate the regular grammars for grammar logics and program expressions of $\mathrm{PDL}^{(0)+id}$.

Theorem 4 [Kle56] Let L be a language on the alphabet Σ. L is a regular language iff there is a regular expression e such that $\mathrm{LAN}(e) = \mathrm{L}$.

Moreover, there is an effective procedure to compute e from a regular grammar $\mathcal{G}$ in exponential-time in $|\mathcal{G}|$ (see e.g. [HU79, Chapter 2]). Given a regular language L, we write e_L to denote a regular expression such that $\mathrm{LAN}(e_\mathrm{L}) = \mathrm{L}$. When L is defined from a regular grammar $\mathcal{G}$, for $i \in N \cup \Sigma$, we write $e_\mathcal{G}(i)$ to denote a regular expression such that $\mathrm{LAN}(e_\mathcal{G}(i)) = \mathrm{L}_i(\mathcal{G})$. Observe that $|e_\mathcal{G}(i)|$ is in $\mathcal{O}(2^{|\mathcal{G}|})$ in the worst case [EZ76] and there are known algorithms to compute some $e_\mathcal{G}(i)$ from $\mathcal{G}$ and i (see e.g. [HU79, Chapter 2]).

3.4 Logics

Let $\mathcal{G}$ be a grammar and L be a modal language such that the set of modal expressions of L includes $\{\mathsf{c}_1, \ldots, \mathsf{c}_m\}$ and $\mathcal{S}$ be a set of L-models. We write $\mathcal{S}^\mathcal{G}$ to denote the subset of $\mathcal{S}$ such that for any $\mathcal{M} \in \mathcal{S}$, $\mathcal{M} \in \mathcal{S}^\mathcal{G}$ $\overset{\text{def}}{\Leftrightarrow}$ for any production rule $i_1 \ldots i_k \to j_1 \ldots j_{k'}$ in $\mathcal{G}$, we have $R_{\mathsf{c}_{j_1}} \circ \ldots \circ$

$R_{\mathsf{C}_{j_{k'}}} \subseteq R_{\mathsf{C}_{i_1}} \circ \ldots \circ R_{\mathsf{C}_{i_k}}$. For the logic $\mathcal{L} = \langle \mathrm{L}, \mathcal{S} \rangle$, we write $\mathcal{L}^{\mathcal{G}}$ to denote the logic $\langle \mathrm{L}, \mathcal{S}^{\mathcal{G}} \rangle$.

For any string $u = i_1 \cdot \ldots \cdot i_n$ in $\{i : i \geq 1\}^*$, we write R_u to denote $R_{\mathsf{C}_{i_1}} \circ \ldots \circ R_{\mathsf{C}_{i_n}}$. When $u = \epsilon$, $R_u \stackrel{\text{def}}{=} Id_W$. For any $u \in \{1, \ldots, m\}^*$, we write $[u]\phi$ to denote the L_m-formula $[\mathsf{c}_{i_1}] \ldots [\mathsf{c}_{i_n}]\phi$ where $u = i_1 \cdot \ldots \cdot i_n$. If $u = \epsilon$, then $[u]\phi$ is simply ϕ.

Definition 5 [FdCP88] Let $\mathcal{G} = \langle N, \Sigma, P, S \rangle$ be a grammar such that $card(N \cup \Sigma) = m \geq 1$. The logic $\mathcal{L}^{\mathcal{G}}_m = \langle \mathrm{L}_m, \mathcal{S}^{\mathcal{G}}_m \rangle$ such that L_m contains m modal constants and $\mathcal{S}^{\mathcal{G}}_m$ is the class of L_m-models such that $i_1 \ldots i_k \rightarrow j_1 \ldots j_{k'} \in P$ implies $R_{\mathsf{C}_{j_1}} \circ \ldots \circ R_{\mathsf{C}_{j_{k'}}} \subseteq R_{\mathsf{C}_{i_1}} \circ \ldots \circ R_{\mathsf{C}_{i_k}}$, is said to be a *grammar logic*.

The $\mathcal{L}^{\mathcal{G}}_m$-models are also noted $\langle W, R_1, \ldots, R_m, V \rangle$, identifying the R_i's with the R_{C_i}'s for $i \in \{1, \ldots, m\}$. Theorem 6 below states equivalent properties between grammar derivations, validity and frame conditions.

Theorem 6 Let $\mathcal{G}$ be a formal grammar. For $u, v \in (N \cup \Sigma)^*$, (I) $u \Rightarrow^*_{\mathcal{G}} v$ iff (II) $[u]\mathrm{p} \Rightarrow [v]\mathrm{p}$ is $\mathcal{L}^{\mathcal{G}}_m$-valid iff (III) for all $\mathcal{L}^{\mathcal{G}}_m$-models $R_v \subseteq R_u$.

The equivalence between (II) and (III) is a classical correspondence result in modal logic theory (see e.g. [Sah75, Ben84]). (I) implies (II) can be proved by induction on the length of the derivation. (II) implies (I) can be shown by easily adapting the proof of [Bal98, Theorem IV.2.1]. Actually, (II) implies (I) can be also proved without any reference to tableaux calculi. Indeed, the proof of [CS94, Theorem 3] allows us to show (II) implies (I) by using the well-known fact that every ordered monoid can be embedded in an ordered monoid of binary relations (see the details in [CS94]). In order to study the grammar logic $\mathcal{L}^{\mathcal{G}}_m$, what is essential is the value of the set P of production rules whereas once P is fixed, the value of the start symbol S and the distribution of the terminal and non terminal symbols are immaterial for $\mathcal{L}^{\mathcal{G}}_m$-satisfiability. Typically, we even allow the unusual situation in the grammar with Σ empty. Hence, we could have equivalently defined the grammar logics from semi Thue systems.

3.5 Problems

Although we shall study the satisfiability complexity of particular grammar logics $\mathcal{L}^{\mathcal{G}}_m$, we are also interested in the satisfiability complexity when

both the grammar $\mathcal{G}$ and the formulae are inputs. Let C be a class of formal grammars (e.g., REG, LLIN, RLIN, LIN, CF). The *general satisfiability problem* GSP(C) for grammar logics in C is defined as follows:

- Inputs: a grammar $\mathcal{G}$ in C and an L_m-formula ϕ;
- Question: Is ϕ $\mathcal{L}_m^{\mathcal{G}}$-satisfiable?

Dually, we define the *general validity problem* GVP(C) for grammar logics in C by asking whether ϕ is $\mathcal{L}_m^{\mathcal{G}}$-valid. Since ϕ is $\mathcal{L}_m^{\mathcal{G}}$-valid iff $\neg\phi$ is not $\mathcal{L}_m^{\mathcal{G}}$-satisfiable, the complexity of the problems GVP(C) and the complexity of the problems GSP(C) are strongly related. The principal object of study in the paper is the problem GSP(REG).

General satisfiability problems easily lead to undecidability as stated below.

Theorem 7 [Bal98, BGM98] GSP(CF) is undecidable and GSP(RLIN) is decidable.

From the decidability proof of GSP(RLIN) in [Bal98, BGM98], one can also conclude that GSP(RLIN) is in **N2EXPTIME**, since it is shown that every $\mathcal{L}_m^{\mathcal{G}}$-satisfiable formula ϕ has a finite $\mathcal{L}_m^{\mathcal{G}}$-model of cardinality $\mathcal{O}(2^{|\mathcal{G}|\times 2^{|\mathcal{G}|}\times|\phi|})$.

Grammatical constraints on models can be also defined for logics whose languages extend L_m. Let $\mathcal{L}$ be a logic such that the set of modal expressions contains the countably infinite set $\{\mathsf{c}_i : i \geq 1\}$ of program constants. The general satisfiability problem for regular logics from $\mathcal{L}$, denoted GSP-REG($\mathcal{L}$) is defined as follows:

- Inputs: a regular grammar $\mathcal{G}$ and an $\mathcal{L}^{\mathcal{G}}$-formula ϕ;
- Question: Is ϕ $\mathcal{L}^{\mathcal{G}}$-satisfiable?

In the sequel we show that GSP-REG(converse-CPDL) and GSP-REG($\mathcal{L}_{\mathbb{N}}$) are decidable.

3.6 A few more properties

We list below a few more properties that can be easily shown by taking advantage of known results from the literature (see e.g. [BRV01] for adequate references).

1. Using the standard translation from modal logic into classical predicate logic, $\mathcal{L}_m^{\mathcal{G}}$-satisfiability [resp. $\mathcal{L}_m^{\mathcal{G}}$-validity] can be linearly reduced to satisfiability [resp. validity] for the fragment of classical predicate logic with at most $max(2, 2\alpha)$ individual variables where α is the maximal length of strings occurring in production rules of $\mathcal{G}$. By Löwenheim-Skolem Theorem for classical predicate logic, we can restrict ourselves to countable $\mathcal{L}_m^{\mathcal{G}}$-models in the sequel.
2. For any decidable class C of formal grammars (e.g. LIN, REG), the set of pairs $\langle \mathcal{G}, \phi \rangle$ such that $\mathcal{G}$ is in C and ϕ is $\mathcal{L}_m^{\mathcal{G}}$-valid is recursively enumerable.
3. A sound and complete Hilbert-style proof system for $\mathcal{L}_m^{\mathcal{G}}$ consists of the axiom schemes (tautologies of propositional calculus, normal axiom scheme) and inference rules (modus ponens and necessitation) for K_m, the modal logic with m independent K modal connectives, plus the axiom schemes $[u]\mathrm{p} \Rightarrow [v]\mathrm{p}$ for $u \to v \in P$. So given a formal grammar, we associate with each production rule a modal axiom scheme. The logics $\mathcal{L}_m^{\mathcal{G}}$ admit other nice proof-theoretical properties (see e.g. [Kra96, Bal98]).
4. Given a decidable class C of formal grammars, if GSP(C) is undecidable, then there exist a grammar $\mathcal{G}_{\mathrm{C}}$ in C and a formula ϕ_{C} such that ϕ_{C} is $\mathcal{L}_m^{\mathcal{G}_{\mathrm{C}}}$-satisfiable and none of the $\mathcal{L}_m^{\mathcal{G}_{\mathrm{C}}}$-models of ϕ_{C} are finite. Indeed, uniformly, one can design a finite axiomatization for $\mathcal{L}_m^{\mathcal{G}_{\mathrm{C}}}$ and by using standard arguments from [Har58] (see also [BRV01, Chapter 6]), the undecidability of GSP(C) entails the existence of such $\mathcal{G}_{\mathrm{C}}$ and ϕ_{C}. Undecidable problems of the form GSP(C) can be found in Section 6.2 and Section 6.3.

4 EXPTIME-completeness of GSP(REG)

In this section, we establish that GSP(REG) is an **EXPTIME**-complete problem. First we show decidability of GSP(REG) by designing a transformation from GSP(REG) into $\mathrm{PDL}^{(0)+id}$-satisfiability. This provides an **2EXPTIME** upper bound for GSP(REG) but more importantly we exhibit a natural relationship between the languages of sentential forms of $\mathcal{G}$ and program expressions in $\mathrm{PDL}^{(0)+id}$ (Lemma 10). In order to get the **EXPTIME** upper bound, the representation of the above mentioned program expressions as finite automata allows to get a polynomial-time

transformation from GSP(REG) into PDL with finite automata (Theorem 16). The **EXPTIME** lower bound for GSP(REG) is by reduction from the global satisfiability problem for the standard modal logic K (see also Section 8).

4.1 Transformation from GSP(REG) into PDL$^{(0)+id}$-satisfiability

Given a grammar $\mathcal{G}$, the least we can do to understand the complexity of $\mathcal{L}_m^{\mathcal{G}}$-satisfiability is to study the language $\mathrm{L}(\mathcal{G})$. Indeed, for $u \in \mathrm{L}(\mathcal{G})$, $[1]\mathrm{p} \Rightarrow [u]\mathrm{p}$ is $\mathcal{L}_m^{\mathcal{G}}$-valid. However, this is not enough since more generally, for $u \Rightarrow_{\mathcal{G}}^* v$, $[u]\mathrm{p} \Rightarrow [v]\mathrm{p}$ is also $\mathcal{L}_m^{\mathcal{G}}$-valid (see Theorem 6). Actually, the key point here is really to study the language of the *sentential forms* of $\mathcal{G}$. For $i \in \Sigma \cup N$, we write $\mathrm{SF}_i(\mathcal{G})$ to denote the language $\{u \in (\Sigma \cup N)^* : i \Rightarrow_{\mathcal{G}}^* u\}$ of sentential forms of $\mathcal{G}$ generated from the symbol i. It is not difficult to design in linear-time in $|\mathcal{G}|$ a grammar $\mathcal{G}_i'$ that generates the language $\mathrm{SF}_i(\mathcal{G})$. Moreover, we can guarantee that if $\mathcal{G}$ is right linear [resp. left linear, linear, context-free] then $\mathcal{G}_i'$ is also right linear [resp. left linear, linear, context-free]. For instance, in the course of the paper, we shall show that there are regular grammars $\mathcal{G}_1$ and $\mathcal{G}_2$ satisfying $\mathrm{L}(\mathcal{G}_1) = \mathrm{L}(\mathcal{G}_2)$ and sharing the same set of terminal and nonterminal symbols such that $\mathcal{L}_m^{\mathcal{G}_1}$-satisfiability is **PSPACE**-complete and $\mathcal{L}_m^{\mathcal{G}_2}$-satisfiability is **EXPTIME**-complete. However, $\mathcal{G}_1$ and $\mathcal{G}_2$ differ essentially in their language of sentential forms. By anticipating a little, we can say that this corresponds to the biggest complexity gap between two regular grammar logics.

In the sequel, $\mathcal{G}$ is assumed to be a context-free grammar such that for $i \in N$, $\mathrm{SF}_i(\mathcal{G})$ is a regular language. This includes the regular grammars for which $e_{\mathrm{SF}_i(\mathcal{G})}$ can be effectively computed (see e.g. [HU79, Chapter 2]). For the other context-free grammars, we assume the existence of $e_{\mathrm{SF}_i(\mathcal{G})}$. There is no way to have a constructive way to obtain $e_{\mathrm{SF}_i(\mathcal{G})}$ since checking whether a linear grammar generates a regular language is an undecidable problem (see e.g. [MS97, page 31]).

Lemma 8 below states a basic property about the languages of sentential forms.

Lemma 8 If $i \to i_1 \cdot \ldots \cdot i_n \in P$, then $\mathrm{SF}_{i_1}(\mathcal{G}) \cdot \ldots \cdot \mathrm{SF}_{i_n}(\mathcal{G}) \subseteq \mathrm{SF}_i(\mathcal{G})$.

Now, let us define a map f from the set of L_m-formulae into the set of PDL$^{(0)+id}$-formulae:

1. $f(\mathrm{p}) \stackrel{\text{def}}{=} \mathrm{p}$ for any propositional variable p;
2. f is homomorphic with respect to the Boolean connectives;
3. $f([i]\phi) \stackrel{\text{def}}{=} [f(e_{\mathrm{SF}_i(\mathcal{G})})]f(\phi)$ for $i \in N \cup \Sigma$;

where f is extended to regular expressions from $\mathcal{E}_{\Sigma \cup N}$ as follows:

4. $f(\epsilon) \stackrel{\text{def}}{=} id \quad f(i) \stackrel{\text{def}}{=} i$ for $i \in \Sigma \cup N$,
5. $f(e_1 \cup e_2) \stackrel{\text{def}}{=} f(e_1) \cup f(e_2) \quad f(e_1 \cdot e_2) \stackrel{\text{def}}{=} f(e_1); f(e_2) \quad f(e^*) \stackrel{\text{def}}{=} f(e)^*$.

The symbols $\cup$ and * are overloaded here. For a fixed grammar $\mathcal{G}$, $f(\phi)$ can be computed in logarithmic space in $|\phi|$. To be precise we should write $f_{\mathcal{G}}$ instead of f but such a subscript is omitted since no confusion will arise.

Example 9 Let $\mathcal{G}$ be the left linear grammar $\mathcal{G} = \langle \{1\}, \{2\}, \{1 \to 1 \cdot 2, 1 \to \epsilon\}, 1 \rangle$. We have $\mathrm{SF}_1(\mathcal{G}) = \{\epsilon, 1\} \cdot \{2\}^*$ and $\mathrm{SF}_2(\mathcal{G}) = \{2\}$. So $e_{\mathrm{SF}_1(\mathcal{G})}$ can take the value $(\epsilon \cup 1) \cdot 2^*$. In the formula map f, every occurrence of the modal connective [2] is replaced by $[\mathsf{c}_2]$ and every occurrence of the modal connective [1] is replaced by $[(id \cup \mathsf{c}_1); \mathsf{c}_2^*]$. In order to be precise one should say that $e_{\mathrm{SF}_1(\mathcal{G})}$ is equivalent to $(\epsilon \cup 1) \cdot 2^*$, but this type of distinction is immaterial in the rest of the paper.

The restriction of the map f to regular expressions simply takes advantage of the fact that any regular expression e over $N \cup \Sigma$ without occurrence of $\emptyset$ can be easily viewed as a syntactic variant of the program expression $f(e)$. The semantical counterpart of this property is stated in Lemma 10.

Lemma 10 Let $\mathcal{M}$ be a $\mathrm{PDL}^{(0)+id}$-model and e be a regular expression in $\mathcal{E}_{\Sigma \cup N}$ such that $\emptyset$ does not occur in e. Then, $R_{f(e)} = \bigcup\{R_u : u \in \mathrm{LAN}(e)\}$.

Proof The proof is by induction on the structure of e.
The base cases $e = \epsilon$, $e = i$ for some $i \in N \cup \Sigma$ are by an easy verification.
Induction step:
Case 1: $e = e_1 \cup e_2$.
$R_{f(e_1 \cup e_2)} = R_{f(e_1) \cup f(e_2)}$ by definition of f and $R_{f(e_1) \cup f(e_2)} = R_{f(e_1)} \cup R_{f(e_2)}$ by definition of the $\mathrm{PDL}^{(0)+id}$ semantics. By the induction hypothesis, for $i \in \{1, 2\}$, $R_{f(e_i)} = \bigcup\{R_u : u \in \mathrm{LAN}(e_i)\}$. Consequently, $R_{f(e_1 \cup e_2)} = \bigcup\{R_u : u \in \mathrm{LAN}(e_1)\} \cup \bigcup\{R_u : u \in \mathrm{LAN}(e_2)\}$. By an

easy manipulation on the union set operator and on the union operator on languages, we get $R_{f(e_1 \cup e_2)} = \bigcup\{R_u : u \in \mathrm{LAN}(e_1 \cup e_2)\}$.
Case 2: $e = e_1 \cdot e_2$.
The proof is similar to Case 1 modulo the adequate obvious modifications.
Case 3: $e = e_1^*$.
$R_{f(e_1^*)} = R_{f(e_1)^*}$ by definition of f and $R_{f(e_1)^*} = R^*_{f(e_1)}$ by definition of the $\mathrm{PDL}^{(0)+id}$ semantics. By the induction hypothesis, $R_{f(e_1)} = \bigcup\{R_u : u \in \mathrm{LAN}(e_1)\}$. Hence, $R^*_{f(e_1)} = \bigcup\{R_{u^1} \circ \ldots \circ R_{u^k} : u^1, \ldots, u^k \in \mathrm{LAN}(e_1)\}$. By definition of the Kleene star operation on languages, we get $R^*_{f(e_1)} = \bigcup\{R_u : u \in \mathrm{LAN}(e_1)^*\}$. This implies that $R_{f(e_1^*)} = \bigcup\{R_u : u \in \mathrm{LAN}(e_1^*)\}$.

Although the class of regular languages is closed under complementation, intersection and reverse operation (see e.g. [Per90]), adding one of these corresponding operations on relations to $\mathrm{PDL}^{(0)+id}$ and to the construction of regular expressions is sufficient to refute Lemma 10. This provides to the operations *, $\cdot$ and $\cup$ a special status among the operations on languages that preserve regularity of languages. These operators are already known to be rather peculiar since they are safe for bisimulation (see e.g. [Ben98b]) unlike intersection, converse and complement.

Lemma 10 allows us to characterize the class of $\mathcal{L}^{\mathcal{G}}_m$-models as substructures of $\mathrm{PDL}^{(0)+id}$-models. This is the object of the two next lemmas. Let $\mathcal{M} = \langle W, (R_\pi)_{\pi \in \Pi}, V\rangle$ be a $\mathrm{PDL}^{(0)+id}$-model. We write $\mathcal{G}(\mathcal{M}) = \langle W, R'_1, \ldots, R'_m, V\rangle$ to denote the L_m-model such that for $i \in N \cup \Sigma$, $R'_i \stackrel{\mathrm{def}}{=} R_{f(e_{\mathrm{SF}_i(\mathcal{G})})}$.

Lemma 11 Let $\mathcal{M}$ be a $\mathrm{PDL}^{(0)+id}$-model. Then $\mathcal{G}(\mathcal{M})$ is an $\mathcal{L}^{\mathcal{G}}_m$-model.

Proof Assume that $i \to i_1 \cdot \ldots \cdot i_n \in P$ from $\mathcal{G}$. We have to show that $R_{f(e_{\mathrm{SF}_{i_1}(\mathcal{G})})} \circ \ldots \circ R_{f(e_{\mathrm{SF}_{i_n}(\mathcal{G})})} \subseteq R_{f(e_{\mathrm{SF}_i(\mathcal{G})})}$. By Lemma 10, $R_{f(e_{\mathrm{SF}_{i_1}(\mathcal{G})})} \circ \ldots \circ R_{f(e_{\mathrm{SF}_{i_n}(\mathcal{G})})} = \bigcup\{R_u : u \in \mathrm{LAN}(e_{\mathrm{SF}_{i_1}(\mathcal{G})})\} \circ \ldots \circ \bigcup\{R_u : u \in \mathrm{LAN}(e_{\mathrm{SF}_{i_n}(\mathcal{G})})\}$. Consequently, $R_{f(e_{\mathrm{SF}_{i_1}(\mathcal{G})})} \circ \ldots \circ R_{f(e_{\mathrm{SF}_{i_n}(\mathcal{G})})} = \bigcup\{R_u : u \in \mathrm{LAN}(e_{\mathrm{SF}_{i_1}(\mathcal{G})}) \cdot \ldots \cdot \mathrm{LAN}(e_{\mathrm{SF}_{i_n}(\mathcal{G})})\}$. This means that $R_{f(e_{\mathrm{SF}_{i_1}(\mathcal{G})})} \circ \ldots \circ R_{f(e_{\mathrm{SF}_{i_n}(\mathcal{G})})} = \bigcup\{R_u : u \in \mathrm{SF}_{i_1}(\mathcal{G}) \cdot \ldots \cdot \mathrm{SF}_{i_n}(\mathcal{G})\}$. By Lemma 8, $\mathrm{SF}_{i_1}(\mathcal{G}) \cdot \ldots \cdot \mathrm{SF}_{i_n}(\mathcal{G}) \subseteq \mathrm{SF}_i(\mathcal{G})$. Hence $R_{f(e_{\mathrm{SF}_{i_1}(\mathcal{G})})} \circ \ldots \circ R_{f(e_{\mathrm{SF}_{i_n}(\mathcal{G})})} \subseteq \bigcup\{R_u : u \in \mathrm{SF}_i(\mathcal{G})\}$ and by Lemma 10, $R_{f(e_{\mathrm{SF}_{i_1}(\mathcal{G})})} \circ \ldots \circ R_{f(\mathrm{SF}_{i_n}(\mathcal{G}))} \subseteq R_{f(e_{\mathrm{SF}_{i_1}(\mathcal{G})})}$.

One can also show the other inclusion.

Lemma 12 $\{\mathcal{G}(\mathcal{M}) : \mathcal{M}$ is a PDL$^{(0)+id}$–model$\}$ is the class of $\mathcal{L}_m^{\mathcal{G}}$-models.

Proof One inclusion is from Lemma 11. Let $\mathcal{M} = \langle W, R'_1, \ldots, R'_m, V\rangle$ be an $\mathcal{L}_m^{\mathcal{G}}$-model. We build an PDL$^{(0)+id}$-model $\mathcal{M}'$ such that $\mathcal{G}(\mathcal{M}') = \mathcal{M}$. Let $\mathcal{M}' = \langle W, (R_\pi)_{\pi\in\Pi}, V\rangle$ be the PDL$^{(0)+id}$-model such that

- for $i \in N \cup \Sigma$, $R_{\mathsf{C}_i} \stackrel{\text{def}}{=} R'_i$; for $i \geq m+1$, $R_{\mathsf{C}_i} \stackrel{\text{def}}{=} \emptyset$ (arbitrary value);
- the relations indexed by constructed program terms are uniquely defined from the definition of the relations by the program constants.

Let us show that for $i \in N \cup \Sigma$, $R_{f(e_{\mathrm{SF}_i(\mathcal{G})})} = R'_i$. This is obvious when $i \in \Sigma$ since in that case $R_{f(e_{\mathrm{SF}_i(\mathcal{G})})} = R_{\mathsf{C}_i}$ and by definition of R_{C_i}, $R_{\mathsf{C}_i} = R'_i$. Now consider the case $i \in N$. By Lemma 10, $R_{f(e_{\mathrm{SF}_i(\mathcal{G})})} = \bigcup\{R_u : u \in \mathrm{SF}_i(\mathcal{G})\}$. Since $i \in \mathrm{SF}_i(\mathcal{G})$, we have $R_{\mathsf{C}_i} \subseteq R_{f(e_{\mathrm{SF}_i(\mathcal{G})})}$. So, $R'_i \subseteq R_{f(e_{\mathrm{SF}_i(\mathcal{G})})}$. Now suppose that $R'_i \subset R_{f(e_{\mathrm{SF}_i(\mathcal{G})})}$ (strict inclusion). So there is $u \in \mathrm{SF}_i(\mathcal{G})$ and $\langle x, y\rangle \in R_{\mathsf{C}_{i_1}} \circ \ldots \circ R_{\mathsf{C}_{i_n}}$ such that $u = i_1 \cdot \ldots \cdot i_n$ and $\langle x, y\rangle \notin R'_i$. Consequently, $i \Rightarrow^*_{\mathcal{G}} u$ and by Theorem 6, $R_{\mathsf{C}_{i_1}} \circ \ldots \circ R_{\mathsf{C}_{i_n}} = R'_{i_1} \circ \ldots \circ R'_{i_n} \subseteq R'_i$, a contradiction. So $R_{f(e_{\mathrm{SF}_i(\mathcal{G})})} = R'_i$. Consequently, $\mathcal{M} = \mathcal{G}(\mathcal{M}')$.

Lemma 12 is interesting not only because it characterizes the class of $\mathcal{L}_m^{\mathcal{G}}$-models but also because it allows us to consider the logic $\mathcal{L}_m^{\mathcal{G}}$ as the fragment of PDL$^{(0)+id}$ restricted to the program expressions $f(e_{\mathrm{SF}_i(\mathcal{G})})$ for $i \in N \cup \Sigma$.

Now we are in position to show that f is satisfiability preserving and this can be done smoothly.

Theorem 13 Let $\mathcal{G}$ be a context-free grammar such that for $i \in N$, $\mathrm{SF}_i(\mathcal{G})$ is a regular language. For any L$_m$-formula ϕ, ϕ is $\mathcal{L}_m^{\mathcal{G}}$-satisfiable iff $f(\phi)$ is PDL$^{(0)+id}$-satisfiable.

Proof ($\rightarrow$) Suppose that ϕ is $\mathcal{L}_m^{\mathcal{G}}$-satisfiable. Let $\langle W, R_1, \ldots, R_m, V\rangle$ be an $\mathcal{L}_m^{\mathcal{G}}$-model and $w \in W$ such that $\mathcal{M}, w \models \phi$. By Lemma 12, there is an PDL$^{(0)+id}$-model $\mathcal{M}' = \langle W, (R_\pi)_{\pi\in\Pi}, V\rangle$ such that for $i \in N \cup \Sigma$, $R_{f(e_{\mathrm{SF}_i(\mathcal{G})})} = R'_i$ which entails that $\mathcal{M}', w \models f(\phi)$.
($\leftarrow$) Suppose that $f(\phi)$ is PDL$^{(0)+id}$-satisfiable. There exist a PDL$^{(0)+id}$-model $\mathcal{M} = \langle W, (R_\pi)_{\pi\in\Pi}, V\rangle$ and $w \in W$ such that $\mathcal{M}, w \models f(\phi)$. By Lemma 11, $\mathcal{G}(\mathcal{M})$ is an $\mathcal{L}_m^{\mathcal{G}}$-model and obviously $\mathcal{G}(\mathcal{M}), w \models \phi$.

By the proof of Theorem 13 and since PDL has the finite model property (see e.g. [HKT00]), $\mathcal{L}_m^{\mathcal{G}}$ has the finite model property and $\mathcal{L}_m^{\mathcal{G}}$-satisfiability is in **EXPTIME**. Additionally, GSP(REG) is in **2EXPTIME**: each grammar $\mathcal{G}$ is an input and therefore each length $|f(e_{\mathrm{SF}_i(\mathcal{G})})|$ is not anymore considered as a constant of the problem. This causes an exponential blow-up due to the fact that in the worst-case, $|f(e_{\mathrm{SF}_i(\mathcal{G})})|$ is in $\mathcal{O}(2^{|\mathcal{G}|})$ [EZ76].

It is natural to wonder what is the relationship between GSP(REG) and PDL$^{(0)+id}$. In fact, for $\alpha \geq 2$, there is no regular grammar $\mathcal{G}$ such that $f(e_{\mathrm{SF}_1(\mathcal{G})}) = \mathsf{c}_1^{\alpha}$. Consequently, GSP(REG) encodes a *proper* subset of program expressions from PDL$^{(0)+id}$ (even if the star operator * is not taken into account). Furthermore, GSP(REG) captures a form of recursion weaker than PDL$^{(0)+id}$ but as will be shown later, this restriction preserves the deterministic exponential-time complexity lower bound. Typically, given a program expression π of PDL$^{(0)+id}$, there may not exist a regular grammar $\mathcal{G}$ and $i \in N \cup \Sigma$ such that $f(e_{\mathrm{SF}_i(\mathcal{G})})$ is equivalent to π. Hence, f cannot be applied backwards in a systematic way.

4.2 A digression on context-free grammars

It is undecidable whether the language generated by a linear grammar is regular (see e.g. [MS97, page 31]) and therefore $e_{\mathrm{SF}_i(\mathcal{G})}$ cannot be always effectively computed. This does not happen when $\mathcal{G}$ is regular. However, there are situations when we can decide whether a context-free grammar generates a regular language (see e.g. [HU79, EHR83]).

Example 14 Let $\mathcal{G} = \langle \{1\}, \emptyset, \{1 \to 1^{k_1}, \ldots, 1 \to 1^{k_n}\}, 1\rangle$ be a context-free grammar for some $n \geq 1$ and $0 \leq k_1 < k_2 < \ldots < k_n$. If $k_1 \geq 1$, then $\mathrm{SF}_1(\mathcal{G}) = \{1\} \cdot \{(1)^{k_1-1}, \ldots, (1)^{k_n-1}\}^*$. If $k_1 = 0$ and $n \geq 2$, then $\mathrm{SF}_1(\mathcal{G}) = \{1\}^*$ otherwise if $k_1 = 0$ and $n = 1$, then $\mathrm{SF}_1(\mathcal{G}) = \{1, \epsilon\}$. $\mathcal{L}_1^{\mathcal{G}}$-satisfiability is in **EXPTIME** and when $k_1 \geq 1$, the map f transforms $[1]$ into $[\mathsf{c}_1; (\mathsf{c}_1^{k_1-1} \cup \ldots \cup \mathsf{c}_1^{k_n-1})^*]$. Observe that for the classical cases $n = 1$, $k_1 = 1$ (modal logic K), $n = 1$, $k_1 = 0$ (modal logic T) $n = 1$, $k_1 = 2$ (modal logic K4), and $n = 2$, $k_1 = 0$, $k_2 = 2$ (modal logic S4), we get uniformly that $[1]$ is translated into $[\mathsf{c}_1; (id)^*]$ (equivalent to $[\mathsf{c}_1]$), $[\mathsf{c}_1 \cup id]$, $[\mathsf{c}_1; \mathsf{c}_1^*]$ and $[\mathsf{c}_1^*]$, respectively.

In the monomodal case, more general decidability results than those in Example 14 exist in the literature. For instance, in [Zak97] it is proved

that any normal extension of S4 by a finite amount of formulae of one variable has the finite model property and is decidable. Observe that the reduction principles of the form $[u]\mathrm{p} \Rightarrow [v]\mathrm{p}$ contain a unique propositional variable.

Example 15 Let $\mathcal{G} = \langle \{1,2\}, \{3\}, \{1 \to 2\cdot 1\cdot 2, 2 \to \epsilon, 2 \to 3, 2 \to 2\cdot 2\}, 1\rangle$ be a context-free grammar. One can show that

$$\mathrm{SF}_1(\mathcal{G}) = \{2,3\}^* \cdot \{1\} \cdot \{2,3\}^* \quad \mathrm{SF}_2(\mathcal{G}) = \{2,3\}^* \quad \mathrm{SF}_3(\mathcal{G}) = \{3\}.$$

The map f transforms [1] [resp. [2], [3]] into $[(\mathsf{c}_2 \cup \mathsf{c}_3)^*; \mathsf{c}_1; (\mathsf{c}_2 \cup \mathsf{c}_3)^*]$ [resp. $[(\mathsf{c}_2 \cup \mathsf{c}_3)^*]$, $[\mathsf{c}_3]$]. Hence, $\mathcal{L}_3^{\mathcal{G}}$-satisfiability is in **EXPTIME**.

It is tempting to extend the previous result for GSP(REG) to a larger class of grammars by appropriately extending PDL. There is a polynomial time transformation from GSP(CF) into satisfiability for PDL with pushdown automata. Indeed, pushdown automata characterizes the class of context-free languages and one can build in polynomial-time a pushdown automata recognizing the language of a context-free grammar (see e.g. [HU79, Chapter 5]). However, the validity problem for PDL with the single addition of the context-free program $a^{\Delta}ba^{\Delta}$ defined as $\bigcup_{i\geq 0} a^i; b; a^i$ is already highly undecidable (Π_1^1-complete) (see e.g. [HKT00, Chapter 9]). In the sequel, we shall also establish that GSP(LIN) is undecidable. Hence, it seems that the regular grammar logics are a very special subset of grammar logics providing an interesting analogy with the place of regular languages among the class of formal languages. Decidable non regular extensions of PDL can be found in [HR93, HKT00].

4.3 Complexity upper and lower bounds

In PDL, regular languages are represented by regular expressions. In PDL with finite automata (APDL) (see e.g. [HKT00]), regular languages are represented by finite nondeterministic automata. APDL can be thought of PDL with finite-state automata taking the place of program expressions (regular expressions). Although the technique to use Kleene's translation from finite automata to regular expressions [Kle56] costs an additional exponential in the worst case [EZ76], APDL-satisfiability is also in **EXPTIME** (see e.g. [HKT00]).

Theorem 16 GSP(REG) is in **EXPTIME**.

Proof Every right linear grammar can be transformed into a finite nondeterministic automaton in polynomial time. Hence, consider the map f previously defined where $f(e_{\mathrm{SF}_i(\mathcal{G})})$ is replaced by the finite automaton $\mathcal{A}_i$ accepting $\mathrm{SF}_i(\mathcal{G})$. Say we get the map f^A. Let $\mathcal{G}'_i$ be a right linear grammar generating $\mathrm{SF}_i(\mathcal{G})$. $\mathcal{G}'_i$ can be built in linear-time in $\mathcal{G}$ and therefore $|\mathcal{G}'_i|$ is in $\mathcal{O}(|\mathcal{G}|)$. $\mathcal{A}_i$ can be computed in polynomial time from $\mathcal{G}'_i$ by roughly associating transitions between states in $\mathcal{A}_i$ to each production rule of $\mathcal{G}'_i$. Using the proof of Theorem 13 one can then show that ϕ is $\mathcal{L}_m^{\mathcal{G}}$-satisfiable iff $f^A(\phi)$ is satisfiable in APDL. Furthermore, $f^A(\phi)$ can be computed in polynomial time in $|\mathcal{G}| + |\phi|$. If $\mathcal{G}$ is a left linear grammar, then let $\mathcal{G}_1$ be the right linear grammar obtained from $\mathcal{G}$ by replacing the production rules $i \to u$ by $i \to u^{-1}$. Obviously, for $i \in N$, $\mathrm{L}_i(\mathcal{G})^{-1} = \mathrm{L}_i(\mathcal{G}_1)$. Now, let $\mathcal{A}_i^{-1}$ be the finite automaton accepting $\mathrm{SF}_i(\mathcal{G}_1)$. The finite automaton $\mathcal{A}_i$ accepting $\mathrm{SF}_i(\mathcal{G})$ is obtained from $\mathcal{A}_i^{-1}$ by swapping the final states and the initial state and by reversing the transitions. Some details are omitted here but this is the essence of the transformation. In all the cases (left or right linear grammar), for $i \in N$, the finite automata $\mathcal{A}_i$ accepting $\mathrm{SF}_i(\mathcal{G})$ can be computed in polynomial time in $|\mathcal{G}|$. Consequently, there is a transformation in polynomial time from GSP(REG) into APDL-satisfiability.

Corollary 17 GVP(REG) and GSP-REG($\mathcal{L}_{\mathbb{N}}$) are in **EXPTIME**.

In order to show that GSP(REG) is **EXPTIME**-hard, it is sufficient to find a regular grammar such that $\mathcal{L}_m^{\mathcal{G}}$-satisfiability is **EXPTIME**-hard.

Lemma 18 Let $\mathcal{G}_0$ be the left linear grammar $\langle\{1\}, \{2\}, \{1 \to 1 \cdot 2\}, 1\rangle$. Then, $\mathcal{L}_2^{\mathcal{G}_0}$-satisfiability is **EXPTIME**-hard.

As far as we know, the logic $\mathcal{L}_2^{\mathcal{G}_0}$ is the simplest extension of K_2 (bimodal logic with two independent K modal connectives) that is **EXPTIME**-hard. Indeed, only the axiom schema $[1]\mathrm{p} \Rightarrow [1][2]\mathrm{p}$ is added to K_2 axiomatization. The proof of Lemma 18 is omitted at this stage of the paper since Lemma 18 is generalized by Theorem 38 for which a detailed proof is given in Section 8.

Theorem 19 GSP(REG), GVP(REG) and GSP-REG($\mathcal{L}_{\mathbb{N}}$) are **EXPTIME**-hard.

Theorem 19 is a consequence of Lemma 18.

5 Other EXPTIME-complete problems

Results in Section 4 can be extended to other logics and problems.

5.1 Global logical consequence and global satisfiability

GSP(REG) and GVP(REG) are **EXPTIME**-complete problems. These results can be lifted to logical consequence and global satisfiability that are also standard problems for modal logics. The general global logical consequence problem for regular logics GGLC(REG) is defined as follows:

- Inputs: a regular grammar $\mathcal{G}$ and two L_m-formulae ϕ, ψ;
- Question: Is it the case that for all $\mathcal{L}_m^{\mathcal{G}}$-models $\mathcal{M}$, $\mathcal{M} \models \phi$ implies $\mathcal{M} \models \psi$?

Similarly, the general global satisfiability problem for regular logics GGSAT(REG) is defined as follows:

- Inputs: a regular grammar $\mathcal{G}$ and an L_m-formula ϕ;
- Question: Is there an $\mathcal{L}_m^{\mathcal{G}}$-model $\mathcal{M}$ such that $\mathcal{M} \models \phi$?

For any regular grammar $\mathcal{G}$, $\mathrm{GLC}(\mathcal{L}_m^{\mathcal{G}})$ [resp. $\mathrm{GSAT}(\mathcal{L}_m^{\mathcal{G}})$] denotes the problem obtained from GGLC(REG) [resp. GGSAT(REG)] by fixing the grammar to $\mathcal{G}$. We write $\mathrm{GLC}(\mathrm{PDL}^{(0)+id})$ [resp. $\mathrm{GSAT}(\mathrm{PDL}^{(0)+id})$] to denote the global logical consequence problem [resp. the global satisfiability problem] for $\mathrm{PDL}^{(0)+id}$.

Theorem 20 GGLC(REG) and GGSAT(REG) are in **EXPTIME**.

Proof Let $\mathcal{G}$ be a regular grammar and ϕ, ψ be L_m-formulae. Using the model constructions from the proof of Theorem 13, one can show that $\langle \phi, \psi \rangle \in \mathrm{GLC}(\mathcal{L}_m^{\mathcal{G}})$ (or equivalently $\langle \mathcal{G}, \phi, \psi \rangle \in \mathrm{GGLC(REG)}$) iff $\langle f(\phi), f(\psi) \rangle \in \mathrm{GLC}(\mathrm{PDL}^{(0)+id})$. Similarly, $\phi \in \mathrm{GSAT}(\mathcal{L}_m^{\mathcal{G}})$ (or equivalently $\langle \mathcal{G}, \phi \rangle \in \mathrm{GGSAT(REG)}$) iff $f(\phi) \in \mathrm{GSAT}(\mathrm{PDL}^{(0)+id})$.

The only program constants occurring in $f(\phi), f(\psi)$ are $\mathsf{c}_1, \ldots, \mathsf{c}_m$. So $\langle f(\phi), f(\psi) \rangle \in \mathrm{GLC}(\mathrm{PDL}^{(0)+id})$ iff $[(\mathsf{c}_1 \cup \ldots \cup \mathsf{c}_m)^*] f(\phi) \Rightarrow f(\psi)$ is $\mathrm{PDL}^{(0)+id}$-valid and $f(\phi) \in \mathrm{GSAT}(\mathrm{PDL}^{(0)+id})$ iff $[(\mathsf{c}_1 \cup \ldots \cup \mathsf{c}_m)^*] f(\phi)$ is $\mathrm{PDL}^{(0)+id}$-satisfiable.

Hence, $\langle \mathcal{G}, \phi, \psi \rangle \in \mathrm{GGLC(REG)}$ iff $[\mathcal{A}] f^A(\phi) \Rightarrow f^A(\psi)$ is APDL-valid and $\langle \mathcal{G}, \phi \rangle \in \mathrm{GGSAT(REG)}$ iff $[\mathcal{A}] f^A(\phi)$ is APDL-satisfiable where

- $\mathcal{A}$ is a finite automaton recognizing $(N \cup \Sigma)^*$ of size $\mathcal{O}(card(N) + card(\Sigma))$;
- $f^A(\phi)$, $f^A(\psi)$ are obtained from $f(\phi)$, $f(\psi)$ respectively by replacing $f(e_{\mathrm{SF}_i(\mathcal{G})})$ by the finite automaton $\mathcal{A}_i$ accepting $\mathrm{SF}_i(\mathcal{G})$ as done in the proof of Theorem 16.

Consequently, GGLC(REG) and GGSAT(REG) are in **EXPTIME**.

This entails that for any regular grammar $\mathcal{G}$, GLC($\mathcal{L}_m^{\mathcal{G}}$) and GSAT($\mathcal{L}_m^{\mathcal{G}}$) are in **EXPTIME**. In order to establish the **EXPTIME** lower bound one can show the following result.

Lemma 21 Let $\mathcal{G}$ be a formal grammar (not necessarily regular). If GSAT($\mathcal{L}_m^{\mathcal{G}}$) is **EXPTIME**-hard, then GLC($\mathcal{L}_m^{\mathcal{G}}$) is also **EXPTIME**-hard.

The proof of Lemma 21 uses the property that GSAT($\mathcal{L}_m^{\mathcal{G}}$) is a subproblem of $\overline{\mathrm{GLC}(\mathcal{L}_m^{\mathcal{G}})}$, the complement problem of GLC($\mathcal{L}_m^{\mathcal{G}}$). Moreover, since **EXPTIME** = co**EXPTIME** [Pap94, Chapter 7], $\overline{\mathrm{GLC}(\mathcal{L}_m^{\mathcal{G}})}$ is **EXPTIME**-hard iff GLC($\mathcal{L}_m^{\mathcal{G}}$) is **EXPTIME**-hard.

Proof Let $\overline{\mathrm{GLC}(\mathcal{L}_m^{\mathcal{G}})}$ be the complement problem of GLC($\mathcal{L}_m^{\mathcal{G}}$), that is $\overline{\mathrm{GLC}(\mathcal{L}_m^{\mathcal{G}})}$ is the set of pairs $\langle \phi, \psi \rangle$ of L_m-formulae such that there is an $\mathcal{L}_m^{\mathcal{G}}$-model $\mathcal{M}$ and w in $\mathcal{M}$ such that $\mathcal{M} \models \phi$ and $\mathcal{M}, w \not\models \psi$. Hence, GSAT($\mathcal{L}_m^{\mathcal{G}}$) is a subproblem of $\overline{\mathrm{GLC}(\mathcal{L}_m^{\mathcal{G}})}$ by taking $\psi = \bot$. Assume that GSAT($\mathcal{L}_m^{\mathcal{G}}$) is **EXPTIME**-hard. This implies that $\overline{\mathrm{GLC}(\mathcal{L}_m^{\mathcal{G}})}$ is also **EXPTIME**-hard. So for any language L in **EXPTIME**, there is a polynomial-time transformation g such that for any finite string $x \in \Sigma_1^*$ (Σ_1 alphabet of L), $x \in$ L iff $g(x) \in \overline{\mathrm{GLC}(\mathcal{L}_m^{\mathcal{G}})}$. Consequently, for any L in **EXPTIME**, there is a **P**-time transformation g such that for any $x \in \Sigma_1^*$, $x \notin \overline{\mathrm{L}}$ ($\overline{\mathrm{L}}$ complement language of L) iff $g(x) \notin \mathrm{GLC}(\mathcal{L}_m^{\mathcal{G}})$. That is, for any L in **EXPTIME**, there is a **P**-time transformation g such that for any string $x \in \Sigma_1^*$, $x \in \overline{\mathrm{L}}$ iff $g(x) \in \mathrm{GLC}(\mathcal{L}_m^{\mathcal{G}})$. Since **EXPTIME** = co**EXPTIME** (see e.g. [Pap94, Chapter 7]), one can conclude that for any L in **EXPTIME**, there is a **P**-time transformation g such that for any string $x \in \Sigma_1^*$, $x \in$ L iff $g(x) \in \mathrm{GLC}(\mathcal{L}_m^{\mathcal{G}})$.

Lemma 21 can be generalized to other deterministic complexity classes but its current form is all that we need here.

Theorem 22 For any regular grammar $\mathcal{G}$, GSAT($\mathcal{L}_m^{\mathcal{G}}$) is **EXPTIME**-hard.

The **EXPTIME** lower bound is established by reducing the global satisfiability problem for either the modal logic K or T into GSAT($\mathcal{L}_m^{\mathcal{G}}$).

Proof Let K-GSAT [resp. T-GSAT] be the set of monomodal formulae ϕ for which there is a [resp. reflexive] Kripke structure $\mathcal{M} = \langle W, R, V \rangle$ satisfying $\mathcal{M} \models \phi$. 'K-GSAT' [resp. 'T-GSAT'] stands for the global satisfiability problem for the standard modal logic K [resp. T] that is known to be **EXPTIME**-hard [CL94, Theorem 1] (see also [Hem96]). As in the proof of Theorem 32, we distinguish two cases according to $\Sigma = \emptyset$ or not.
Case 1: $k' \in \Sigma$.
Let us define a logarithmic space transformation from K-GSAT into GSAT($\mathcal{L}_m^{\mathcal{G}}$). Indeed, one can show that $\phi \in$ K-GSAT iff $\phi' \in$ GSAT($\mathcal{L}_m^{\mathcal{G}}$) where ϕ' is obtained from ϕ by replacing every occurrence of $\Box$ by $[k']$. The details are similar to Case 1 in the proof of Theorem 32.
Case 2: $\Sigma = \emptyset$.
As in Case 2 in the proof of Theorem 32, let $k' \in X_{min}$ and $k' \Rightarrow_{\mathcal{G}}^* \epsilon$. One can show that $\phi \in$ T-GSAT iff $\phi' \in$ GSAT($\mathcal{L}_m^{\mathcal{G}}$) where ϕ' is obtained from ϕ by replacing every occurrence of $\Box$ by $[k']$.

We know that as far as satisfiability is concerned, there exist regular grammar logics that are **PSPACE**-complete (the standard modal logic K for instance) and there exist regular grammar logics that are **EXPTIME**-complete (see Lemma 18). Such a dichotomy does not exist for the global logical consequence and global satisfiability.

Corollary 23 (I) For any regular grammar $\mathcal{G}$, GLC($\mathcal{L}_m^{\mathcal{G}}$) and GSAT($\mathcal{L}_m^{\mathcal{G}}$) are **EXPTIME**-complete.
(II) GGLC(REG) and GGSAT(REG) are **EXPTIME**-complete.

In Corollary 23(I), the assumption on the regularity of $\mathcal{G}$ is essential. For instance, the global logical consequence problem for the context-free grammar logic S4 is in **PSPACE** (see e.g. [ABM00, Mas00]) whereas the global satisfiability problem for S4 is "only" **NP**-complete [TK91].

5.2 PDL-like logics

The map f from L_m-formulae into $\mathrm{PDL}^{(0)+id}$-formulae defined in Section 4.1 can be generalized to more expressive source logics while preserving the very essence of f: replace every occurrence of c_i, $i \in N \cup \Sigma$, by $f(e_{\mathrm{SF}_i(\mathcal{G})})$ and the other syntactic objects in program expressions remain unchanged.

We give some examples below although other extensions of $\mathrm{PDL}^{(0)+id}$ could be treated similarly (modal μ-calculus, intersection-PDL, ...). This is omitted here to avoid the boredom of repetitive arguments.

Theorem 24 Let $\mathcal{G}$ be a regular grammar. converse-$\mathrm{CPDL}^{\mathcal{G}}$-satisfiability is **EXPTIME**-complete.

The proof of Theorem 24 contains a logarithmic space transformation from converse-$\mathrm{CPDL}^{\mathcal{G}}$-satisfiability into converse-CPDL-satisfiability.

Proof Roughly speaking, the modal connectives $[\mathsf{c}_{m+1}]$ and $[\mathsf{c}^*_{m+1}]$ behave independently of the modal connectives involving exclusively the program constants $\mathsf{c}_1, \ldots, \mathsf{c}_m$. By [Spa93], converse-$\mathrm{CPDL}^{\mathcal{G}}$-satisfiability is **EXPTIME**-hard. converse-CPDL is in **EXPTIME** [Gia95, ABM00] (see also [PT91]). Let us define a logarithmic space transformation f' from converse-$\mathrm{CPDL}^{\mathcal{G}}$-satisfiability into converse-CPDL-satisfiability. f' is defined as f from $\mathcal{L}^{\mathcal{G}}_m$ into $\mathrm{PDL}^{(0)+id}$ except that

- the point 1. is replaced by: for any atomic formula p (either propositional variable or nominal) $f(\mathrm{p}) \stackrel{\text{def}}{=} \mathrm{p}$.
- the points 3., 4. and 5. are replaced by: $f'([\pi]\phi) \stackrel{\text{def}}{=} [f'(\pi)]f'(\phi)$ where f' is extended as follows:
  1. $f'(\mathrm{U}) \stackrel{\text{def}}{=} \mathrm{U}$;
  2. for $i \in N \cup \Sigma$, $f'(\mathsf{c}_i) \stackrel{\text{def}}{=} f''(e_{\mathrm{SF}_i(\mathcal{G})})$ with
     (a) $f''(j) \stackrel{\text{def}}{=} \mathsf{c}_j$ for $j \in \Sigma \cup N$ $\quad f''(e \cup e') \stackrel{\text{def}}{=} f''(e) \cup f''(e')$,
     (b) $f''(e^*) \stackrel{\text{def}}{=} f''(e)^*$ $\quad f''(e \cdot e') \stackrel{\text{def}}{=} f''(e); f''(e')$,
  3. for $i > m$, $f'(\mathsf{c}_i) \stackrel{\text{def}}{=} \mathsf{c}_i$;
  4. $f'(e_1 \cup e_2) \stackrel{\text{def}}{=} f'(e_1) \cup f'(e_2)$ $\quad f'(e_1^{-1}) \stackrel{\text{def}}{=} f'(e_1)^{-1}$,
  5. $f'(e_1; e_2) \stackrel{\text{def}}{=} f'(e_1); f'(e_2)$,
  6. $f'(e^*) \stackrel{\text{def}}{=} f'(e)^*$ $\quad f'(\varphi?) \stackrel{\text{def}}{=} f'(\varphi)?$.

One can show that ϕ is converse-CPDL$^{\mathcal{G}}$-satisfiable iff $f'(\phi)$ is converse-CPDL-satisfiable.

The converse operator $^{-1}$ cannot be used to express grammatical constraints between relations (see the remarks after Lemma 10) but we can use it to build constructed program expressions. The proof of Theorem 24 also provides an exponential-time transformation from REG-GSP(converse-CPDL) into converse-CPDL satisfiability. Since converse-CPDL satisfiability is in **EXPTIME** [ABM00] we obtain the following result. As a corollary, REG-GSP(converse-CPDL) is in **2EXPTIME**.

By combining the proof of Theorem 24 (restricted to PDL$^{(0)}$ only) and the proof of Theorem 16, one can easily show the following result.

Theorem 25 REG-GSP(PDL$^{(0)}$) is **EXPTIME**-complete.

Proof REG-GSP(PDL$^{(0)}$) is **EXPTIME**-hard because GSP(REG) is **EXPTIME**-hard and GSP(REG) is a subproblem of REG-GSP(PDL$^{(0)}$). Since PDL$^{(0)}$ is a fragment of converse-CPDL, REG-GSP(PDL) is also in **2EXPTIME**. However, this upper bound is not sharp enough. Actually, REG-GSP(PDL$^{(0)}$) can be transformed in polynomial time to APDL-satisfiability which provides the required **EXPTIME** upper bound for REG-GSP(PDL$^{(0)}$). Observe that any program expression of PDL$^{(0)}$ can be transformed in polynomial time to a finite automaton recognizing the same language (see e.g. [HU79, Chapter 2]). We introduce a slight change in such a process for building finite automata from regular expressions: for $i \in N \cup \Sigma$, each occurrence of c_i is replaced by the finite automata $\mathcal{A}_i$ recognizing the language generated by $f(e_{\mathrm{SF}_i(\mathcal{G})})$ (this can be done in polynomial time in $\mathcal{G}$, see also the proof of Theorem 13).

5.3 Description logics with inclusion axioms

The standard description logic $\mathcal{ALC}$ (see e.g. [SSS91, DLNN97]) is known to be a syntactic variant of the multimodal logic $\mathcal{L}_{\mathbb{N}}$ [Sch91]. Corollary 17 can be reformulated in the DLs lingua. Let $\mathcal{ALCC}omp$ be the extension of $\mathcal{ALC}$ by adding the composition operator for *atomic roles*. A *role hierarchy* is defined as a finite set of expressions of the form $R \sqsubseteq S$ where R and S are (composed) roles (see e.g. [HS99, HM00]). An expression of the form $R \sqsubseteq S$ is also known as a role value inclusion in the DL literature (with composed atomic roles). With each role inclusion axiom $R \sqsubseteq S$, we associate a production rule $S \to R$. The satisfiability problem for $\mathcal{ALCC}omp$ with respect to a role hierarchy is to

determine given a concept and a role hierarchy whether the concept has a nonempty interpretation verifying the role hierarchy. A consequence of Corollary 17 and Theorem 19 is the following: the satisfiability problem for $\mathcal{ALCC}comp$ with role hierarchies defining a set of production rules from a regular grammar is **EXPTIME**-complete. In Section 6, we show that the satisfiability problem for $\mathcal{ALCC}comp$ with role hierarchies defining a set of production rules from a linear grammar is undecidable. The above **EXPTIME**-completeness characterization is however more robust when considering extensions of $\mathcal{ALC}$. Let $\mathcal{C}^{(0)}$ be the description logic (see e.g. [GL94]) corresponding to PDL$^{(0)}$. We get rid of the test operator '?' for the sake of simplicity. A reformulation of Theorem 25 is the following: the satisfiability problem for $\mathcal{C}^{(0)}$ with role hierarchies defining a set of production rules from a regular grammar is **EXPTIME**-complete.

6 Undecidability of GSP(LIN)

Concept subsumption in the description logic $\mathcal{AL}$ with role value maps admitting composition of atomic roles is known to be undecidable [SS89] by reduction from the undecidable word problem for semigroups. Besides, in [Bal98, BGM98], it is shown that GSP(CF) is undecidable by reducing the problem of empty intersection between context-free languages into GSP(CF). The emptiness problem for the intersection of linear languages is also known to be undecidable [RS94, Theorem 2.11]. Using the proof technique from [Bal98, BGM98], one can show that GSP(LIN) is also undecidable. This proof technique relies on the completeness of a prefixed tableaux calculus for grammar logics. In the sequel, we propose an alternative proof that only relies on the characterization of context-free languages. Namely, context-free languages are minimal solutions of a system of equations the solutions for which are obtainable by means of successive approximations, starting from the empty set (see e.g. [ABB97]). Our proof below illustrates once more that although it is doubtful whether grammar logics are useful to analyze formal languages, formal languages theory can provide insight into issues for multimodal logics.

6.1 The syntactic $\mathcal{L}_m^{\mathcal{G}}$-frame

In the rest of this section, $\mathcal{G} = \langle N, \Sigma, P, S\rangle$ denotes a context-free grammar such that $N \cup \Sigma = \{1, \ldots, m\}$, $N = \{1, \ldots, k\}$, $k < m$ and $S = 1$.

For $i \in \{1, \ldots, k\}$, there is $l_i \geq 0$ such that $i \to u_{i,1}, \ldots, i \to u_{i,l_i}$ are the only production rules in P having i as left-hand side.

Let W be a nonempty countable set and $R_{k+1}, \ldots, R_m$ be binary relations on W. Let $g : \mathcal{P}(W \times W)^k \to \mathcal{P}(W \times W)^k$ be the map such that $g(R_1, \ldots, R_k) \stackrel{\text{def}}{=} \langle \bigcup_{1 \leq j \leq l_1} R_{u_{1,j}}, \ldots, \bigcup_{1 \leq j \leq l_k} R_{u_{k,j}} \rangle$. To be more precise, one should subscript g by $R_{k+1}, \ldots, R_m$. This is omitted for the sake of clarity. Let $\leq$ be the binary relation on $\mathcal{P}(W \times W)^k$ defined as follows: $\langle R_1, \ldots, R_k \rangle \leq \langle R'_1, \ldots, R'_k \rangle \stackrel{\text{def}}{\Leftrightarrow}$ for all $i \in \{1, \ldots, k\}$, $R_i \subseteq R'_i$. The structure $\langle \mathcal{P}(W \times W)^k, \leq \rangle$ is a complete lattice and g is continuous and order-preserving. By Kleene's Theorem, the least fixed point of g exists and is equal to

$$\mu(g) = \bigcup_{i \in \mathbb{N}} g^i(\emptyset, \ldots, \emptyset) \stackrel{\text{def}}{=} \langle \mathcal{R}_1, \ldots, \mathcal{R}_k \rangle.$$

Observe that for $i \in \{1, \ldots, k\}$, $\mathcal{R}_i \subseteq (R_{k+1} \cup \ldots \cup R_m)^*$.

Now, let us build the *syntactic* $\mathcal{L}^{\mathcal{G}}_m$-frame $\mathcal{F}_{syn} = \langle W_{syn}, R_{1,syn}, \ldots, R_{m,syn} \rangle$:

- $W_{syn} \stackrel{\text{def}}{=} \{k+1, \ldots, m\}^*$;
- $u R_{i,syn} u' \stackrel{\text{def}}{\Leftrightarrow} u' = u \cdot i$ for some $i \in \{k+1, \ldots, m\}$, $u, u' \in W_{syn}$;
- $\langle R_{1,syn}, \ldots, R_{k,syn} \rangle$ is the least fixed point of g defined from W_{syn}, $R_{k+1,syn}, \ldots, R_{m,syn}$.

Consequently,

- $(\bigcup_{j \in \Sigma} R_{j,syn})^*(\epsilon) = W_{syn}$ and for $i \in N$, $R_{i,syn} \subseteq (\bigcup_{j \in \Sigma} R_{j,syn})^*$;
- for $u, v \in W_{syn}$, $R_u(v) = \{v \cdot u\}$.

The subscript 'syn' is omitted when no confusion can arise. Obviously, $\mathcal{F}_{syn} = \langle W_{syn}, R_{k+1,syn}, \ldots, R_{m,syn} \rangle$ is an $(m-k)$-ary infinite tree. $\mathcal{F}_{sync}$ is a canonical structure for $\mathcal{L}^{\mathcal{G}}_m$ since not only it is an $\mathcal{L}^{\mathcal{G}}_m$-frame but it encodes explicitly the languages $\mathrm{L}_i(\mathcal{G})$ for $i \in N$ (see Lemma 26 below).

Lemma 26 For $i \in N$, for $u \in W_{syn}$, $u \in R_i(\epsilon)$ iff $u \in \mathrm{L}_i(\mathcal{G})$.

The proof of Lemma 26 is based on the property that the context-free languages $\mathrm{L}_1(\mathcal{G}), \ldots, \mathrm{L}_k(\mathcal{G})$ are minimal solutions of the system of equations defined from the context-free grammar $\mathcal{G}$.

Proof Let $g' : [\{k+1,\ldots,m\}^*]^k \to [\{k+1,\ldots,m\}^*]^k$ be the map such that

$$g'(\mathrm{L}_1,\ldots,\mathrm{L}_k) \stackrel{\text{def}}{=} \langle \bigcup_{1\leq j\leq l_1} u'_{1,j},\ldots,\bigcup_{1\leq j\leq l_k} u'_{k,j}\rangle$$

where $u'_{i,j}$ is obtained from $u_{i,j}$ by replacing the elements of $j' \in \Sigma$ by the language $\{j'\}$ and the elements of $i' \in N$ by $\mathrm{L}_{i'}$. Here, the concatenation operator $\cdot$ acts on languages. Let $\leq'$ be the binary relation on $[\{k+1,\ldots,m\}^*]^k$ defined as follows: $\langle \mathrm{L}_1,\ldots,\mathrm{L}_k\rangle \leq' \langle \mathrm{L}'_1,\ldots,\mathrm{L}'_k\rangle \stackrel{\text{def}}{\Leftrightarrow}$ for all $i \in \{1,\ldots,k\}$, $\mathrm{L}_i \subseteq \mathrm{L}'_i$. The structure $\langle [\{k+1,\ldots,m\}^*]^k, \leq'\rangle$ is a complete lattice and g' is continuous and order-preserving. By Kleene's Theorem, the least fixed point of g' exists and is equal to $\mu(g') = \bigcup_{i\in\mathbb{N}} g'^i(\emptyset,\ldots,\emptyset)$. By Schützenberger's Theorem [Sch62], for $u \in \{k+1,\ldots,m\}^*$, for $i \in \{1,\ldots,k\}$, $u \in \mathrm{L}_i(\mathcal{G})$ iff $u \in \mu(g')[i]$ (ith element of the tuple $\mu(g')$). Indeed, the context-free languages $\mathrm{L}_1(\mathcal{G})$, $\ldots$, $\mathrm{L}_k(\mathcal{G})$ are minimal solutions of the system of equations defined from the context-free grammar $\mathcal{G}$. By an induction on i, we show that for $i \geq 0$, for $j \in \{1,\ldots,k\}$, $g^i(\emptyset,\ldots,\emptyset)[j] = \bigcup\{R_u : u \in g'^i(\emptyset,\ldots,\emptyset)[j]\}$.
Base case 1: $i = 0$.
$g^i(\emptyset,\ldots,\emptyset) = \langle\emptyset,\ldots,\emptyset\rangle$ and $g'^i(\emptyset,\ldots,\emptyset) = \langle\emptyset,\ldots,\emptyset\rangle$. Hence, $g^i(\emptyset,\ldots,\emptyset)[j] = \emptyset = \bigcup\{R_u : u \in \emptyset\} = \bigcup\{R_u : u \in g'^i(\emptyset,\ldots,\emptyset)[j]\}$.
Base case 2: $i = 1$.
$g(\emptyset,\ldots,\emptyset)[j] = \bigcup_{j'\in\{1,\ldots,l_j\},u_{j,j'}\in\Sigma^*} R_{u_{j,j'}}$. Besides, $g'(\emptyset,\ldots,\emptyset)[j] = \bigcup\{u_{j,j'} : j' \in \{1,\ldots,l_j\}, u_{j,j'} \in \Sigma^*\}$. So $g(\emptyset,\ldots,\emptyset)[j] = \bigcup\{R_u : u \in g'(\emptyset,\ldots,\emptyset)[j]\}$.
Induction step:
By definition of g^{i+1}, $g(g^i(\emptyset,\ldots,\emptyset))[j] = \bigcup_{j'\in\{1,\ldots,l_j\}} R_{u'_{j,j'}}$ where $R_{u'_{j,j'}}$ is obtained from $R_{u_{j,j'}}$ by replacing for $\alpha \in \{1,\ldots,k\}$, R_α by $g^i(\emptyset,\ldots,\emptyset)[\alpha]$. Similarly, $g'(g'^i(\emptyset,\ldots,\emptyset))[j] = \bigcup_{j'\in\{1,\ldots,l_j\}} u''_{j,j'}$ where $u''_{j,j'}$ is obtained from $u_{j,j'}$ by replacing for $\alpha \in \{1,\ldots,k\}$, α by the language $g'^i(\emptyset,\ldots,\emptyset)[\alpha]$ and for $\alpha \in \{k+1,\ldots,m\}$, α by the language $\{\alpha\}$.

Let (i) $\langle x,y\rangle \in g^{i+1}(\emptyset,\ldots,\emptyset)[j]$. (i) holds true iff there is $j' \in \{1,\ldots,l_j\}$ such that $\langle x,y\rangle \in R_{u'_{j,j'}}$. Assume that $u_{j,j'}$ has the form $u_{j,j'} = w_1\cdot i_1\cdot w_2\cdot i_2\ldots\cdot i_l\cdot w_{l+1}$ with $i_1,\ldots,i_l \in \{1,\ldots,k\}$ and $w_1,\ldots,w_{l+1} \in \Sigma^*$. Hence, (i) iff (ii) $\langle x,y\rangle \in R_{w_1}\circ g^i(\emptyset,\ldots,\emptyset)[i_1]\circ\ldots\circ g^i(\emptyset,\ldots,\emptyset)[i_l]\circ R_{w_{l+1}}$. Furthermore, (ii) iff there is a finite family $(x_i)_{i\in\{0,\ldots,2\times l+1\}}$ such that

- $x_0 = x$; $x_{2\times l+1} = y$;
- for $s \in \{1,\ldots,l\}$, $\langle x_{2\times s}, x_{2\times s+1}\rangle \in R_{w_{s+1}}$;
- for $s \in \{0,\ldots,l-1\}$, $\langle x_{2\times s+1}, x_{2\times s+2}\rangle \in g^i(\emptyset,\ldots,\emptyset)[i_{s+1}]$.

By the induction hypothesis, for $s \in \{0, \ldots, l-1\}$,

$$\langle x_{2\times s+1}, x_{2\times s+2}\rangle \in \bigcup\{R_{u'} : u' \in g^{'i}(\emptyset, \ldots, \emptyset)[i_{s+1}]\}$$

This means that for $s \in \{0, \ldots, l-1\}$, there is $u'_s \in g^{'i}(\emptyset, \ldots, \emptyset)[i_{s+1}]$ such that $\langle x_{2\times s+1}, x_{2\times s+2}\rangle \in R_{u'_s}$. Hence, (ii) iff $\langle x, y\rangle \in R_{w_1} \circ R_{u'_1} \circ \ldots \circ R_{u'_l} \circ R_{w_{l+1}}$ for some $u'_1, \ldots, u'_l$ such that for $s \in \{0, \ldots, l-1\}$, $u'_s \in g^{'i}(\emptyset, \ldots, \emptyset)[i_{s+1}]$. However, precisely, $w_1 \cdot u'_1 \cdot \ldots \cdot u'_l \cdot w_{l+1} \in g^{'i+1}(\emptyset, \ldots, \emptyset)[j]$. Consequently, (ii) iff there is $u \in g^{'i+1}(\emptyset, \ldots, \emptyset)[j]$ such that $\langle x, y\rangle \in R_u$.

This entails that for $u \in W_{syn}$, for $i \in \{1, \ldots, k\}$, $u \in R_{i,syn}(\epsilon)$ iff $u \in \mathrm{L}_i(\mathcal{G})$. Indeed, suppose that $u \in R_{i,syn}(\epsilon)$. So there is $\alpha \in \mathbb{N}$ such that $\langle \epsilon, u\rangle \in g^{\alpha}(\emptyset, \ldots, \emptyset)[i]$. By the above result proved by induction, this is equivalent to the existence of some $v \in g^{'\alpha}(\emptyset, \ldots, \emptyset)[i]$ such that $\langle \epsilon, u\rangle \in R_v$. However both u and v are in Σ^*, so by construction of the syntactic frame, $u = v$ and therefore $u \in g^{'\alpha}(\emptyset, \ldots, \emptyset)[i]$. By Schützenberger's Theorem, this implies $u \in \mathrm{L}_i(\mathcal{G})$. The proof for the other direction is similar

6.2 The empty intersection problem reduced to GSP(LIN)

The ideas behind the reduction given below are from [Bal98, BGM98] and therefore the modal encoding below is given in order to make the paper self-contained. However, the proof differs in its use of the syntactic frames instead of tableaux proofs.

In order to encode the empty intersection question for two context-free grammars $\mathcal{G}_1$ and $\mathcal{G}_2$, we introduce a third grammar $\mathcal{G}$ for which $\mathcal{L}_m^{\mathcal{G}}$-satisfiability can express the empty intersection question. Basically, $\mathcal{G}$ contains both $\mathcal{G}_1$ and $\mathcal{G}_2$ with the additional ability to generate all the words of the terminal alphabet. Let $\mathcal{G}_1 = \langle\{1, \ldots, k\}, \{k{+}1, \ldots, m\}, P_1, 1\rangle$, $\mathcal{G}_2 = \langle\{1, \ldots, k\}, \{k+1, \ldots, m\}, P_2, 1\rangle$ be context-free grammars with $k+1 \leq m$. Let $\mathcal{G} = \langle N', \Sigma', P'_1 \cup P'_2 \cup P', 1\rangle$ be the grammar such that

1. $N' \stackrel{\text{def}}{=} \{1, \ldots, 2\times k+1\}$; $\Sigma' \stackrel{\text{def}}{=} \{2\times k+2, \ldots, m+k+1\}$;
2. $P' \stackrel{\text{def}}{=} \{1 \to i, 1 \to 1\cdot i : i \in \Sigma'\}$;
3. For $u \to v \in P_1$ we write $u_1 \to v_1$ to denote the production rule in $\mathcal{G}$ obtained by replacing each $i \in \{1, \ldots, k\}$ by $i+1$ and each $j \in \{k+1, \ldots, m\}$ by $j+k+1$. $P'_1 \stackrel{\text{def}}{=} \{u_1 \to v_1 : u \to v \in P_1\}$.

4. For $u \to v \in P_2$ we write $u_2 \to v_2$ to denote the production rule in $\mathcal{G}$ obtained by replacing each $i \in \{1, \ldots, k\}$ by $i + k + 1$ and each $j \in \{k + 1, \ldots, m\}$ by $j + k + 1$. $P_2' \stackrel{\text{def}}{=} \{u_2 \to v_2 : u \to v \in P_2\}$.

Observe that if $\mathcal{G}_1, \mathcal{G}_2$ are both context-free [resp. linear], then $\mathcal{G}$ is context-free [resp. linear]. $\mathcal{G}$ is actually obtained from $\mathcal{G}_1$ and $\mathcal{G}_2$ by adding a nonterminal symbol (namely 1), $\{2, \ldots, k+1\}$ in N' corresponds to $\{1, \ldots, k\}$ in N_1 and $\{k + 2, \ldots, 2 \times k + 1\}$ in N' corresponds to $\{1, \ldots, k\}$ in N_2. Roughly speaking, Σ' corresponds to the alphabet of $\mathcal{G}_1$ [resp. to the alphabet of $\mathcal{G}_2$] by operating a shift of $k + 1$. The non terminal symbol 1 in $\mathcal{G}$ generates all the words in Σ'^*. Moreover, $\mathrm{L}_2(\mathcal{G}) = \{u+(k+1) : u \in \mathrm{L}(\mathcal{G}_1)\}$ and $\mathrm{L}_{k+2}(\mathcal{G}) = \{u+(k+1) : u \in \mathrm{L}(\mathcal{G}_2)\}$. The string $u+(k+1)$ is obtained from u by adding $k+1$ to each element of u. Let $\phi_\cap$ be the L_{m+k+1}-formula below:

$$\phi_\cap \stackrel{\text{def}}{=} (\bigwedge_{i \in \Sigma'} (\langle i \rangle \top \wedge [1]\langle i \rangle \top)) \Rightarrow ([2]\mathrm{p} \Rightarrow \langle k + 2 \rangle \mathrm{p})$$

The condition $\bigwedge_{i \in \Sigma'} (\langle i \rangle \top \wedge [1]\langle i \rangle \top)$ is satisfied in an L_{m+k+1}-model at x iff all words in Σ'^* can be found starting at x.

Lemma 27 (I) $\phi_\cap$ is $\mathcal{L}^{\mathcal{G}}_{m+k+1}$-valid iff (II) $\mathrm{L}(\mathcal{G}_1) \cap \mathrm{L}(\mathcal{G}_2) \neq \emptyset$.

In order to prove that (I) implies (II), Lemma 26 is used in some essential way.

Proof (II) $\to$ (I) Assume that $u_0 \in \mathrm{L}(\mathcal{G}_1) \cap \mathrm{L}(\mathcal{G}_2)$. Hence, $[1]\mathrm{p} \Rightarrow [u_0]\mathrm{p}$ is both $\mathcal{L}^{\mathcal{G}_1}_m$-valid and $\mathcal{L}^{\mathcal{G}_2}_m$-valid and, $[2]\mathrm{p} \Rightarrow [u_0+(k+1)]\mathrm{p}$ and $[k+2]\mathrm{p} \Rightarrow [u_0+(k+1)]\mathrm{p}$ are $\mathcal{L}^{\mathcal{G}}_{m+k+1}$-valid. $u_0+(k+1)$ is obtained from u_0 by adding $k+1$ to each element of u_0. Let $\mathcal{M} = \langle W, R_1, \ldots, R_m, V \rangle$ be an $\mathcal{L}^{\mathcal{G}}_{m+k+1}$-model and $w \in W$. Assume that $\mathcal{M}, w \models (\bigwedge_{i \in \Sigma'} (\langle i \rangle \top \wedge [1]\langle i \rangle \top)) \wedge [2]\mathrm{p}$. So for $u \in \Sigma'^*$, $R_u(w) \neq \emptyset$ and $R_{u_0+(k+1)} \subseteq R_2$. Hence, there is $w' \in R_{u_0+(k+1)}(w)$ such that $\mathcal{M}, w' \models \mathrm{p}$. Furthermore, $[k+2]\mathrm{p} \Rightarrow [u_0+(k+1)]\mathrm{p}$ is $\mathcal{L}^{\mathcal{G}}_{m+k+1}$-valid which implies $R_{u_0+(k+1)} \subseteq R_{k+2}$ (by Theorem 6). So $\mathcal{M}, w \models \langle k + 2 \rangle \mathrm{p}$.
(I) $\to$ (II) Assume that $\phi_\cap$ is $\mathcal{L}^{\mathcal{G}}_{m+k+1}$-valid. Let $\mathcal{M}$ be an $\mathcal{L}^{\mathcal{G}}_{m+k+1}$-model $\langle W_{syn}, R_{1,syn}, \ldots, R_{m+k+1,syn}, V \rangle$ based on the syntactic frame of $\mathcal{L}^{\mathcal{G}}_{m+k+1}$ such that $V(\mathrm{p}) \stackrel{\text{def}}{=} R_{2,syn}(\epsilon)$. By construction, $\mathcal{M}, \epsilon \models \bigwedge_{i \in \Sigma'} (\langle i \rangle \top \wedge [1]\langle i \rangle \top)$ and $\mathcal{M}, \epsilon \models [2]\mathrm{p}$. By validity of the formula $\phi_\cap$, $\mathcal{M}, \epsilon \models \langle k+2 \rangle \mathrm{p}$. Hence, there is $u_0 \in R_{k+2,syn}(\epsilon)$ such that $\mathcal{M}, u_0 \models \mathrm{p}$. By Lemma 26,

$u_0 \in \mathrm{L}_{k+2}(\mathcal{G})$ which is equivalent to $u_0^- \in \mathrm{L}(\mathcal{G}_2)$ where u_0^- is obtained from u_0 by replacing each occurrence of $j \in \Sigma'$ by $j - k - 1$. By definition of $V(\mathrm{p})$, $u_0 \in R_{2,syn}(\epsilon)$ since $\mathcal{M}, u_0 \models \mathrm{p}$, which is equivalent to $u_0^- \in \mathrm{L}(\mathcal{G}_1)$ by Lemma 26. Hence, $u_0^- \in \mathrm{L}(\mathcal{G}_1) \cap \mathrm{L}(\mathcal{G}_2)$.

Theorem 28 GSP(LIN) is undecidable.

Indeed, any two linear grammars $\mathcal{G}_1$ and $\mathcal{G}_2$ with nonempty sets of terminal symbols can be isomorphically put into the above form. Since the problem of empty intersection between linear grammars is undecidable, by Lemma 27, GVP(LIN) is undecidable and equivalently GSP(LIN) is undecidable.

The undecidability of GSP(LIN) provides a partial negative answer to the decidability status of the modal μ-calculus with systems of relation fixed points [Ben98a, Section 4]. Hence, expressing recursion over binary relations in a modal logic even though restricted to the present linear case, leads to undecidability. By contrast, by adequately adapting the proof of Theorem 24, GSP-REG(modal μ-calculus) can be shown to be decidable. Regularity might be a line of attack to show decidability of modal logics.

6.3 Other undecidable subproblems of GSP(CF)

The technique to prove that GSP(LIN) is undecidable can be extended to other classes of context-free grammars, even if the length of the right-hand sides of the production rules is at most two. Let CHOM be the class of context-free grammars $\mathcal{G}$ in Chomsky normal form, that is for any production rule $i \to u$ of $\mathcal{G}$, $u \in N^2 \cup \Sigma$. Similarly, let 2CF be the class of context-free grammars $\mathcal{G}$ such that for any production rule $i \to u$ of $\mathcal{G}$, $u \in N^2 \cup \{i \cdot i : i \in \Sigma\}$. We consider GSP(CHOM) and GSP(2CF) in this work since GSP(2CF) is closely related to a fragment of a description logic with role inclusion axioms including those of the form $S \circ T \sqsubseteq R$, that is mentioned in [WHM00, Section 4]. The grammars in CHOM are particularly interesting since for any context-free grammar $\mathcal{G}$ such that $\epsilon \notin \mathrm{L}(\mathcal{G})$, one can effectively construct a grammar $\mathcal{G}'$ in CHOM such that $\mathrm{L}(\mathcal{G}) = \mathrm{L}(\mathcal{G}')$ (see e.g. [HU79, Section 4.5]). Consequently, it is not hard to show that the problem of empty intersection between context-free grammars in CHOM is also undecidable.

Theorem 29 GSP(CHOM) is undecidable.

Proof Given $\mathcal{G}_1$, $\mathcal{G}_2$ in CHOM, let $\mathcal{G}$ be the context-free grammar defined as in Section 6.2 except that P' takes the value $\{1 \to 1 \cdot 1\} \cup \{1 \to i : i \in \Sigma'\}$. As in the proof of Lemma 27, one can show that $\phi_\cap$ is $\mathcal{L}^{\mathcal{G}}_{m+k+1}$-valid iff $\mathrm{L}(\mathcal{G}_1) \cap \mathrm{L}(\mathcal{G}_2) \neq \emptyset$. Since the constructed $\mathcal{G}$ is also in CHOM, this entails the undecidability of GSP(CHOM).

Using a small trick one can also show that GSP(2CF) is undecidable.

Theorem 30 GSP(2CF) is undecidable.

Proof Let $\mathcal{G} = \langle N, \Sigma, P, S\rangle$ be in CHOM. We write $\mathrm{double}(\mathcal{G}) = \langle N, \Sigma, P', S\rangle$ to denote the grammar in 2CF such that

$$P' \stackrel{\text{def}}{=} \{i \to u \in P : |u| = 2\} \cup \{i \to j \cdot j : i \to j \in P, j \in \Sigma\}.$$

It is easy to show that $\mathrm{L}(\mathrm{double}(\mathcal{G})) = \{i_1 \cdot i_1 \cdot \ldots i_n \cdot i_n : i_1 \cdot \ldots \cdot i_n \in \mathrm{L}(\mathcal{G})\}$. Since the words of $\mathrm{L}(\mathrm{double}(\mathcal{G}))$ are obtained from those of $\mathrm{L}(\mathcal{G})$ by copying twice each terminal symbol, for any $\mathcal{G}_1$, $\mathcal{G}_2$ in CHOM, $\mathrm{L}(\mathrm{double}(\mathcal{G}_1)) \cap \mathrm{L}(\mathrm{double}(\mathcal{G}_2)) = \emptyset$ iff $\mathrm{L}(\mathcal{G}_1) \cap \mathrm{L}(\mathcal{G}_2) = \emptyset$.

Let $\mathcal{G}_1$ and $\mathcal{G}_2$ be in CHOM and $\mathcal{G}$ be the context-free grammar defined as in Section 6.2 from $\mathrm{double}(\mathcal{G}_1)$ and $\mathrm{double}(\mathcal{G}_2)$ except that P' takes the value $\{1 \to 1 \cdot 1\} \cup \{1 \to i \cdot i : i \in \Sigma'\}$. Let $\phi_\cap$ be the L_{m+k+1}-formula below:

$$\phi_\cap \stackrel{\text{def}}{=} (\bigwedge_{i \in \Sigma'} (\langle i\rangle\langle i\rangle\top \wedge [1]\langle i\rangle\langle i\rangle\top)) \Rightarrow ([2]\mathrm{p} \Rightarrow \langle k+2\rangle \mathrm{p})$$

As in the proof of Lemma 27, one can show that $\phi_\cap$ is $\mathcal{L}^{\mathcal{G}}_{m+k+1}$-valid iff $\mathrm{L}(\mathrm{double}(\mathcal{G}_1)) \cap \mathrm{L}(\mathrm{double}(\mathcal{G}_2)) \neq \emptyset$. Consequently, $\phi_\cap$ is $\mathcal{L}^{\mathcal{G}}_{m+k+1}$-valid iff $\mathrm{L}(\mathcal{G}_1) \cap \mathrm{L}(\mathcal{G}_2) \neq \emptyset$ and $\mathcal{G}$ is in 2CF, which entails the undecidability of GSP(2CF).

By the point 4. in Section 3.6, we obtain the following result.

Corollary 31 For any class C in {LIN, CHOM, 2CF}, there exist a grammar $\mathcal{G}$ in C and a formula ϕ such that ϕ is $\mathcal{L}^{\mathcal{G}}_m$-satisfiable and none of the $\mathcal{L}^{\mathcal{G}}_m$-models of ϕ are finite.

7 PSPACE-complete grammar logics

The multimodal logic K_m, has a **PSPACE**-complete satisfiability problem (see e.g. [HM92]). Adding a regular set of modal axioms preserves the **PSPACE** lower bound.

Theorem 32 Let $\mathcal{G}$ be either a regular grammar or a context-free grammar such that $\Sigma \neq \emptyset$. Then, $\mathcal{L}_m^{\mathcal{G}}$-satisfiability is **PSPACE**-hard.

Proof We distinguish two cases according to $\Sigma = \emptyset$ or not.
Case 1: $k' \in \Sigma$.
Let us define a logarithmic space many-one reduction f from satisfiability for the modal logic K into $\mathcal{L}_m^{\mathcal{G}}$-satisfiability. K-satisfiability is known to be **PSPACE**-complete [Lad77]. $f(\phi)$ is obtained from ϕ by replacing every occurrence of $\Box$ by $[k']$. Suppose there is an $\mathcal{L}_m^{\mathcal{G}}$-model $\mathcal{M} = \langle W, R_1, \ldots, R_m, V\rangle$ and $w \in W$ such that $\mathcal{M}, w \models g(\phi)$. Obviously, $\langle W, R_{k'}, V\rangle, w \models \phi$. Now, suppose that there is a K-model $\mathcal{M} = \langle W, R, V\rangle$ and $w \in W$ such that $\mathcal{M}, w \models \phi$. By using the least fixed point of g in Section 6, we can conclude that there is an $\mathcal{L}_m^{\mathcal{G}}$-model $\mathcal{M}' = \langle W, R_1, \ldots, R_m, V\rangle$ such that $R_{k'} = R$. Moreover, $\mathcal{M}', w \models f(\phi)$. The only difficulty here is the existence of the $\mathcal{L}_m^{\mathcal{G}}$-model such that $R_{k'} = R$.
Case 2: $\Sigma = \emptyset$.
All the production rules in P are either of the form $i \to j$ or of the form $i \to \epsilon$ for some $i, j \in N$. Let $\approx$ be the restriction of $\Rightarrow_{\mathcal{G}}^* \cap (\Rightarrow_{\mathcal{G}}^*)^{-1}$ to N. $\approx$ is obviously an equivalence relation on N. There is an equivalence class X_{min} of $\approx$ such that for all $i \in X_{min}$, there is no $j \in N \setminus X_{min}$ such that $i \to j \in P$. Since X_{min} is nonempty, take an arbitrary k' in X_{min}. If for some $i \in X_{min}$, $i \to \epsilon$, then we define a logarithmic space transformation f from the modal logic T into $\mathcal{L}_m^{\mathcal{G}}$-satisfiability. Otherwise, we define a logarithmic space transformation f' from the modal logic K into $\mathcal{L}_m^{\mathcal{G}}$-satisfiability. T-satisfiability is also known to be **PSPACE**-complete [Lad77]. Actually f and f' are identical. $f(\phi)$ is obtained from ϕ by replacing every occurrence of $\Box$ by $[k']$. Now suppose that for some $i \in X_{min}$, $i \to \epsilon$. The other case is omitted to avoid the boredom of repetitive arguments. Suppose there is an $\mathcal{L}_m^{\mathcal{G}}$-model $\mathcal{M} = \langle W, R_1, \ldots, R_m, V\rangle$ and $w \in W$ such that $\mathcal{M}, w \models f(\phi)$. Obviously, $\langle W, R_{k'}, V\rangle, w \models \phi$ and $R_{k'}$ is reflexive since $k' \Rightarrow_{\mathcal{G}}^* \epsilon$. Now, suppose that there is a T-model $\mathcal{M} = \langle W, R, V\rangle$ and $w \in W$ such that

$\mathcal{M}, w \models \phi$. Let $\mathcal{M}' = \langle W', R'_1, \ldots, R'_m, V' \rangle$ be the $\mathcal{L}^{\mathcal{G}}_m$-model defined as follows:

- $W' \stackrel{\text{def}}{=} W$; $V' \stackrel{\text{def}}{=} V$; for $i \in X_{min}$, $R_i \stackrel{\text{def}}{=} R$;
- for $j \in N \setminus X_{min}$, if $j \Rightarrow^*_{\mathcal{G}} k'$, then $R'_j \stackrel{\text{def}}{=} R$ otherwise $R_j \stackrel{\text{def}}{=} Id_W$.

One can show that for $i, j \in N$, $Id_W \subseteq R_i$ and if $i \Rightarrow^*_{\mathcal{G}} j$, then $R_j \subseteq R_i$. Hence, $\mathcal{M}'$ is really an $\mathcal{L}^{\mathcal{G}}_m$-model and obviously $\mathcal{M}', w \models f(\phi)$.

Since for any regular grammar $\mathcal{G}$ defining a regular grammar logic, $\mathcal{L}^{\mathcal{G}}_m$-satisfiability is **PSPACE**-hard and in **EXPTIME**, the most relevant question about the computational complexity of $\mathcal{L}^{\mathcal{G}}_m$ is to know whether the problem is in **PSPACE** or **EXPTIME**-hard (unless **PSPACE** = **EXPTIME**). The rest of the paper is dedicated to characterize either **PSPACE** regular grammar logics or **EXPTIME**-hard regular grammar logics.

Lemma 33 Let $\mathcal{G}$ be a context-free grammar such that for $i \in N$, $\mathrm{SF}_i(\mathcal{G})$ is finite. Then $\mathcal{L}^{\mathcal{G}}_m$-satisfiability is in **PSPACE**.

Proof Assume that for $i \in N$, $\mathrm{SF}_i(\mathcal{G})$ is finite. Observe that for $i \in \Sigma$, $\mathrm{SF}_i(\mathcal{G}) = \{i\}$ and therefore $\mathrm{SF}_i(\mathcal{G})$ is also finite. For $i \in N$, the regular expression $e_{\mathrm{SF}_i(\mathcal{G})}$ in $\mathcal{E}_{\Sigma \cup N}$ can be the finite union of regular expressions of the form either ϵ or $i_1 \cdot \ldots \cdot i_n$ with $\{i_1, \ldots, i_n\} \subseteq \Sigma$. Hence, the map f from Section 4.1 transforms L_m-formulae into $\mathrm{PDL}^{(0)+id}$-formulae without the star operator *, without converse $^{-1}$ and with no union operator $\cup$ in the scope of the composition operator ;. One can show that this very fragment of $\mathrm{PDL}^{(0)+id}$ has a satisfiability problem in **PSPACE**.

The condition for $i \in N$, $\mathrm{SF}_i(\mathcal{G})$ is finite implies that $\mathrm{L}(\mathcal{G})$ is finite but the converse is generally not true. It is decidable whether a context-free grammar generates an infinite language and the problem is **P**-complete [JL76, Gol81]. So it is tractable to check whether the languages of sentential forms generated by a context-free grammar are all finite.

Lemma 33 does not imply that the general satisfiability problem for the class $\mathrm{REG}^{\mathrm{sf-fin}}$ of regular grammars such that for $i \in N$, $\mathrm{SF}_i(\mathcal{G})$ is finite, is in **PSPACE** since $e_{\mathrm{SF}_i(\mathcal{G})}$ may be in exponential size in $|\mathcal{G}|$ even when $\mathcal{G} \in \mathrm{REG}^{\mathrm{sf-fin}}$. However one can show the following result.

Lemma 34 Let $\mathcal{G}$ be a regular grammar such that for $i \in N$, $\mathrm{L}_i(\mathcal{G})$ is finite. Then, for $i \in N$, one can compute in polynomial time in $|\mathcal{G}|$ a regular expression $e_i \in \mathcal{E}_\Sigma$ such that $\mathrm{LAN}(e_i) = \mathrm{L}_i(\mathcal{G})$.

The proof of Lemma 34 takes advantage of the fact that $\mathrm{L}_i(\mathcal{G})$ can be generated by an acyclic finite automaton.

Proof Assume that $\mathcal{G}$ is right linear. The case when $\mathcal{G}$ is left linear is omitted here but it is very similar. Without any loss of generality, we can assume that for $i, j \in N$, $i \Rightarrow_{\mathcal{G}}^{*} j$ and $j \Rightarrow_{\mathcal{G}}^{*} i$ imply $i = j$. Any regular grammar can be put in that form at a polynomial-time cost while preserving the generated languages (but not the languages of sentential forms).

Let $G = \langle N, R \rangle$ be the directed graph whose nodes are the nonterminal symbols of $\mathcal{G}$ such that for $i, j \in N$, $iRj \stackrel{\text{def}}{\Leftrightarrow} i \to u \cdot j \in P$ for some P in $\mathcal{G}$ ($u \in \Sigma^*$).

Let $i_0 \in N$ be a nonterminal symbol for which we wish to compute e_{i_0}. Let N_{i_0} be the following subset of N: $R^*(i_0) \cap \{i \in N : iR^*j, j \to u \in P, u \in \Sigma^*\}$. Since $\mathrm{L}_{i_0}(\mathcal{G})$ is finite, the restriction of G to N_{i_0} is a DAG. The passage to a restrictive graph allows to get rid of the non productive nonterminal symbols. The set N_{i_0} can be computed in polynomial-time in $\mathcal{G}$. If N_{i_0} is empty, then $e_{i_0} \stackrel{\text{def}}{=} \emptyset$ (regular expression interpreted as the empty language). Otherwise, let $\alpha \geq 1$ be the unique natural number such that $R^{\alpha}(i_0) \neq \emptyset$ and $R^{\alpha+1}(i_0) = \emptyset$. Let us define the partition $X_0, \ldots, X_{\alpha}$ of N_{i_0} in the following way:

- $X_0 \stackrel{\text{def}}{=} \{j \in N_{i_0} : R(j) = \emptyset\}$;
- for $s \in \{0, \ldots, \alpha - 1\}$, $X_s \stackrel{\text{def}}{=} \{j \in N_{i_0} : jRj', j' \in X_{s-1}\}$.

Consequently, $X_{\alpha} = \{i_0\}$. Since the restriction of G to N_{i_0} is a DAG, and by construction of α, $X_0, \ldots, X_{\alpha}$ can be checked to be a partition of N_{i_0}. Now, let us define for $j \in N_{i_0}$, the regular expression e_j:

- for $j \in X_0$, $e_j \stackrel{\text{def}}{=} \bigcup\{u : j \to u \in P\}$;
- for $s \in \{1, \ldots, \alpha\}$, for $j \in X_s$,

$$e_j \stackrel{\text{def}}{=} \bigcup\{u : j \to u \in P\} \bigcup$$

$$\bigcup\{(\bigcup\{u : j \to u \cdot j' \in P\}) \cdot e_{j'} : j' \in X_{s-1}, j \to u' \cdot j' \in P\}.$$

One can check that the whole process to compute e_{i_0} can be done in polynomial time in $|\mathcal{G}|$.

Example 35 For $n \geq 2$, let $\mathcal{G}_n$ be the right linear grammar

$$\langle \{1,\ldots,n\}, \{n{+}1, n{+}2\}, \{j \to (n{+}1){\cdot}(j{+}1), j \to (n{+}2){\cdot}(j{+}1) : 1 \leq j \leq n{-}1\}, 1\rangle$$

The cardinality of $\mathrm{SF}_1(\mathcal{G}_n)$ is in $\mathcal{O}(2^n)$ but the regular expression for $e_{\mathrm{SF}_1(\mathcal{G}_n)}$ computed from the proof of Lemma 34 is $1 \cup (n+1 \cup n+2) \cdot (2 \cup (n+1 \cup n+2) \cdot (3 \ldots ((n-1) \cup (n+1 \cup n+2) \cdot n)))$.

Theorem 36 $\mathrm{GSP}(\mathrm{REG}^{\mathrm{sf-fin}})$ is **PSPACE**-complete.

Proof **PSPACE**-hardness is from Theorem 32. **PSPACE**-easiness can be shown by taking the proof of Lemma 33 and by considering in the map into the star-free fragment of $\mathrm{PDL}^{(0)+id}$ the regular expressions from Lemma 34.

The **PSPACE**-easiness of $\mathrm{GSP}(\mathrm{REG}^{\mathrm{sf-fin}})$ is reminiscent to the **PSPACE**-easiness of the satisfiability of concepts with respect to acyclic terminologies in $\mathcal{ALC}$ [Lut99] (see also the renaming technique in [Min88]). Acyclicity is however related to concepts in [Lut99] whereas it is related to roles in our current investigation.

Besides, Theorem 36 does not exhaust all the possibilities of regular grammar logics in **PSPACE**. There exist regular grammar logics such that for some $i \in N$, $\mathrm{SF}_i(\mathcal{G})$ is infinite (that is $f(e_{\mathrm{SF}_i(\mathcal{G})})$ necessarily contains an occurrence of *) and $\mathcal{L}_m^{\mathcal{G}}$-satisfiability is still in **PSPACE**. This is the subject of the companion paper [Dem01] where **PSPACE** upper bounds are established by proof-theoretical means. Let $\mathrm{RLIN}^{\mathrm{fin}}$ [resp. $\mathrm{LLIN}^{\mathrm{fin}}$] be the class of right [resp. left] linear grammars such that for $i \in N$, $\mathrm{L}_i(\mathcal{G})$ is finite (unlike the grammars in $\mathrm{REG}^{\mathrm{sf-fin}}$, $\mathrm{SF}_i(\mathcal{G})$ is allowed to be infinite). Then,

Theorem 37 [Dem01] $\mathrm{GSP}(\mathrm{RLIN}^{\mathrm{fin}})$ is **PSPACE**-complete.

By contrast, $\mathrm{GSP}(\mathrm{LLIN}^{\mathrm{fin}})$ is **EXPTIME**-complete by Lemma 18 and Theorem 16.

8 Conditions for EXPTIME-hardness

In this section, we shall identify sufficient conditions to guarantee that the satisfiability problem of a regular grammar logic is **EXPTIME**-hard. While Section 8.1 deals with left linear grammars, Section 8.2 deals with

right linear grammars. Such a separation between regular grammars is due to an asymmetry as far as **EXPTIME**-hardness is concerned. However, we identify decidable fragments of such sufficient conditions in Section 8.3.

All the **EXPTIME**-hardness proof in Section 8 are obtained by reducing the global satisfiability problem for the modal logic K (similar to the essence of some proofs in [Spa93, Sat96]). Let K-GSAT be the set of monomodal formulae ϕ for which there is a Kripke structure $\mathcal{M} = \langle W, R, V \rangle$ satisfying $\mathcal{M} \models \phi$. 'K-GSAT' stands for the global satisfiability problem for the standard modal logic K that is known to be **EXPTIME**-hard [CL94, Theorem 1] (see also [Hem96]).

8.1 Left linear grammars

Let $\mathcal{G}$ be a left linear grammar such that $card(N \cup \Sigma) \geq 2$.

Theorem 38 Assume that $i \Rightarrow_{\mathcal{G}}^{*} i \cdot u^{k'}$ for some $i \in N$, $k' \geq 1$, $u = i_1 \cdot \ldots \cdot i_n$, $n \geq 1$ and $j \in \{1, \ldots, n\}$ such that i_j occurs only once in u. Then, $\mathcal{L}_m^{\mathcal{G}}$-satisfiability is **EXPTIME**-hard.

Proof Let us define a logarithmic space transformation from K-GSAT into $\mathcal{L}_m^{\mathcal{G}}$-satisfiability. Let ϕ be a monomodal formula. Let $g(\phi)$ be the formula $\langle i \rangle \top \wedge \bigwedge_{0 \leq \alpha \leq k'-1} [i][u^{\alpha}]\phi'$ where ϕ' is obtained from the formula ϕ by replacing every occurrence of $\Box$ by $[u]$. Let us show that (i) ϕ belongs to K-GSAT iff (ii) $g(\phi)$ is $\mathcal{L}_m^{\mathcal{G}}$-satisfiable.
(i) $\rightarrow$ (ii) Assume that ϕ belongs to K-GSAT. Hence, there is a Kripke model $\mathcal{M} = \langle W, R, V \rangle$ such that $\mathcal{M} \models \phi$. Without any loss of generality, we can assume that $W = R^{*}(w_0)$ for some $w_0 \in W$.

We shall define an $\mathcal{L}_m^{\mathcal{G}}$-model $\mathcal{M}' = \langle W', R_1, \ldots, R_m, V' \rangle$ such that $W' = W$ and $V' = V$. To do so, we introduce an auxiliary PDL$^{(0)+id}$-frame $\langle W, (R'_{\pi})_{\pi \in \Pi} \rangle$ defined as follows:

- $R'_{i_j} \stackrel{\text{def}}{=} R$ and for $l \in (N \cup \Sigma) \setminus \{i_j\}$, $R'_{\mathsf{c}_l} \stackrel{\text{def}}{=} Id_W$;
- for any program constant different from c_l for some $l \in N \cup \Sigma$, R'_{c_l} takes an arbitrary value, say the empty set;
- the relations R'_{π} indexed by constructed program terms are uniquely defined from the definition of the relations by the program constants.

Now we are in position to assign values to R_l for $l \in N \cup \Sigma$: $R_l \stackrel{\text{def}}{=} R'_{f(e_{\mathrm{SF}_l(\mathcal{G})})}$. Since for $l \in \Sigma \setminus \{i_j\}$, $R_l = Id_W$ and by assumption i_j occurs only once in u, we have $R_{i_j} = R_u = R$. By Lemma 11, $\mathcal{M}'$ is an $\mathcal{L}_m^{\mathcal{G}}$-model and $R_i \circ (\bigcup_{\alpha \in \{0,\ldots,k'-1\}} R_u^\alpha) \subseteq W \times W$. Since $R_u^*(w_0) = W$ and $\langle w_0, w_0 \rangle \in R_i$, we have $\mathcal{M}', w_0 \models \langle i \rangle \top \wedge \bigwedge_{0 \leq \alpha \leq k'-1} [i][u^\alpha]\phi'$.
(ii) $\rightarrow$ (i) Assume that $\langle i \rangle \top \wedge \bigwedge_{0 \leq \alpha \leq k'-1} [i][u^\alpha]\phi'$ is $\mathcal{L}_m^{\mathcal{G}}$-satisfiable. So there is an $\mathcal{L}_m^{\mathcal{G}}$-model $\mathcal{M} = \langle W, R_1, \ldots, R_m, V \rangle$ and $w \in W$ such that $\mathcal{M}, w \models \langle i \rangle \top \wedge \bigwedge_{0 \leq \alpha \leq k'-1} [i][u]^\alpha \phi'$. We have $R_i \circ (R_u^{k'})^* \subseteq R_i$. More generally, for $\alpha \in \{0, \ldots, k'-1\}$, we have $R_i \circ (R_u^{k'})^* \circ R_u^\alpha = R_i \circ R_u^\alpha \circ (R_u^{k'})^* \subseteq R_i \circ R_u^\alpha$. Hence, $R_i \circ (R_u)^* \subseteq R_i \circ (\bigcup_{\alpha \in \{0,\ldots,k'-1\}} R_u^\alpha)$. There is $w_0 \in R_i(w)$ and for all $w' \in R_u^*(w_0)$, $\mathcal{M}, w' \models \phi'$. Let $\mathcal{M}' = \langle W', R', V' \rangle$ be the Kripke structure such that $W' \stackrel{\text{def}}{=} R_u^*(w_0)$, R' is the restriction of R_u to W' and V' is the restriction of V to W'. For all $w \in W'$, $\mathcal{M}', w \models \phi$.

In Theorem 38, the existence of some terminal symbol i_j occurring exactly once in u is required for technical reasons (see the part (i) $\rightarrow$ (ii) of the proof of Theorem 38). It is an open question whether Theorem 38 can be extended by allowing any $u \in \Sigma^+$.

Example 39 For $m \geq 2$, $i \geq 1$, let $\mathcal{G}_{m,i}$ be $\langle \{1\}, \{2, \ldots, m\}, \{1 \rightarrow 1 \cdot 2^i\}, 1 \rangle$. By Theorem 38, $\mathcal{L}_m^{\mathcal{G}_{m,i}}$-satisfiability is **EXPTIME**-hard. For distinct prime numbers i, i', $\mathcal{L}_m^{\mathcal{G}_{m,i}}$-satisfiability and $\mathcal{L}_m^{\mathcal{G}_{m,i'}}$-satisfiability are distinct problems. Moreover, for any left linear grammar of the form $\mathcal{G}^+ = \langle \{1\}, \{2, \ldots, m\}, \{1 \rightarrow 1 \cdot 2^i\} \cup P^+, 1 \rangle$, $\mathcal{L}_m^{\mathcal{G}^+}$-satisfiability is also **EXPTIME**-hard.

The condition for the existence of the nonterminal symbol i_j in Theorem 38 can be replaced by another condition as done in Theorem 40 below. This requires another construction of $\mathcal{L}_m^{\mathcal{G}}$-models.

Theorem 40 Assume that $i \Rightarrow_{\mathcal{G}}^* i \cdot u$ such that

- $i \in N$; $u = i_1 \cdot \ldots \cdot i_n$; $n \geq 2$;
- for all $l \in \{1, \ldots, n\}$, i_l occurs at least twice in u;
- $\mathrm{LAN}(u^* \cdot (i_1 \cup (i_1 \cdot i_2) \cup \ldots \cup (i_1 \cdot \ldots \cdot i_{n-1}))) \cap \mathrm{LAN}(e_{\mathcal{G}'}(i)) = \emptyset$ where $\mathcal{G}'$ is $\mathcal{G}$ augmented with the production rule $i \rightarrow \epsilon$.

Then, $\mathcal{L}_m^{\mathcal{G}}$-satisfiability is **EXPTIME**-hard.

Proof Without any loss of generality we can assume that the set N of non terminal symbols of $\mathcal{G}$ is $\{1,\ldots,k\}$ for some $k < m$. Let us define a logarithmic space transformation from K-GSAT into $\mathcal{L}_m^{\mathcal{G}}$-satisfiability. Let ϕ be a monomodal formula. Let $g(\phi)$ be the formula $\langle i\rangle\top \wedge [i]\phi'$ where ϕ' is obtained from the formula ϕ by replacing every occurrence of $\Box$ by $[u]$. Let us show that (i) ϕ belongs to K-GSAT iff (ii) $g(\phi)$ is $\mathcal{L}_m^{\mathcal{G}}$-satisfiable.
(i) $\rightarrow$ (ii) Assume that ϕ belongs to K-GSAT. Hence, there is a Kripke structure $\mathcal{M} = \langle W, R, V\rangle$ such that $\mathcal{M} \models \phi$. Without any loss of generality, we can assume that $W = R^*(w_0)$ for some $w_0 \in W$.

We shall define an $\mathcal{L}_m^{\mathcal{G}}$-model $\mathcal{M}' = \langle W', R_1, \ldots, R_m, V'\rangle$ such that $W' \stackrel{\text{def}}{=} W \cup (R\times\{1,\ldots,n-1\})$, V' restricted to W is V and V' is arbitrarily defined for $R\times\{1,\ldots,n-1\}$. The motivation to introduce W' is to encode xRy in $\mathcal{M}$ by the sequence $x\ R_1''\ \langle\langle x,y\rangle,1\rangle\ R_2''\ \langle\langle x,y\rangle,2\rangle \ldots\ R_{n-1}''\ \langle\langle x,y\rangle, n-1\rangle\ R_n''\ y$ where $R_1'',\ldots,R_n''$ are auxiliary binary relations. Formally, $R_1'',\ldots,R_n''$ are defined as follows:

- $R_1'' \stackrel{\text{def}}{=} \{\langle x,\langle\langle x,y\rangle,1\rangle\rangle : \langle x,y\rangle \in R\}$;
- $R_n'' \stackrel{\text{def}}{=} \{\langle\langle\langle x,y\rangle, n-1\rangle, y\rangle : \langle x,y\rangle \in R\}$;
- for $j \in \{2,\ldots,n-1\}$, $R_j'' \stackrel{\text{def}}{=} \{\langle\langle\langle x,y\rangle, j-1\rangle, \langle\langle x,y\rangle, j\rangle\rangle : \langle x,y\rangle \in R\}$.

Now it remains to relate the R_i''s with the R_is. Typically, if the jth element occurring in u is l then we enforce $R_j'' \subseteq R_l$. To do so, we introduce an auxiliary PDL$^{(0)+id}$-frame $\langle W, (R'_\pi)_{\pi\in\Pi}\rangle$ defined as follows:

- for $l \in N \setminus \{i\}$, $R'_{\mathsf{c}_l} \stackrel{\text{def}}{=} \emptyset$; $R'_{\mathsf{c}_i} \stackrel{\text{def}}{=} Id_W$;
- for $l \in \Sigma$, $R'_{\mathsf{c}_l} \stackrel{\text{def}}{=} \bigcup\{R_j'' : i_j = l, j \in \{1,\ldots,n\}\}$ (remember $u = i_1 \cdot \ldots \cdot i_n$);
- for any program constant different from c_l for some $l \in N \cup \Sigma$, R'_{c_l} takes an arbitrary value, say the empty set;
- the relations R'_π indexed by constructed program terms are uniquely defined from the definition of the relations by the program constants.

Now we are in position to assign values to R_l for $l \in N \cup \Sigma$: $R_l \stackrel{\text{def}}{=} R'_{f(e_{\mathrm{SF}_l(\mathcal{G})})}$. $\mathcal{M}'$ is an $\mathcal{L}_m^{\mathcal{G}}$-model by Lemma 11. By construction, for $w \in W \subseteq W'$, $R_u(w) = R(w)$ and $\langle w_0, w_0\rangle \in R_i$. Additionally, LAN$(u^* \cdot$

$(i_1 \cup (i_1 \cdot i_2) \cup \ldots \cup (i_1 \cdot \ldots \cdot i_{n-1}))) \cap \mathrm{LAN}(e_{\mathcal{G}'}(i)) = \emptyset$ implies that for $w \in W \subseteq W'$, $R_i(w) \subseteq W$.

Let us prove this latter property. Assume that $\mathrm{LAN}(u^* \cdot (i_1 \cup (i_1 \cdot i_2) \cup \ldots \cup (i_1 \cdot \ldots \cdot i_{n-1}))) \cap \mathrm{LAN}(e_{\mathcal{G}'}(i)) = \emptyset$ and suppose that there exist $w' \in W$ and $w'' \in W' \setminus W$ such that $\langle w', w'' \rangle \in R_i$. By Definition of R_i and by Lemma 10, we have

$$R_i = R'_{f(e_{\mathrm{SF}_i(\mathcal{G})})} = \{R'_u : u \in \mathrm{SF}_i(\mathcal{G})\}.$$

Equivalently, $R_i = \{R'_u : u \in \mathrm{SF}_i(\mathcal{G}')\}$ by imposing $R'_{\mathsf{C}_i} = Id_W$. Since for $l \in N \setminus \{i\}$, we already have that $R'_{\mathsf{C}_l} = \emptyset$, this entails that $R_i = \{R'_u : u \in \mathrm{SF}_i(\mathcal{G}') \cap \Sigma^*\}$ by imposing $R'_{\mathsf{C}_i} = Id_W$. Consequently, $R_i = \{R'_u : u \in \mathrm{LAN}(e_{\mathcal{G}'}(i))\}$. So there is $v \in \mathrm{LAN}(e_{\mathcal{G}'}(i))$ such that $w' R'_v w''$. Since $w' \in W$ and $w'' \in W' \setminus W$, we also have that $v \in \mathrm{LAN}(u^* \cdot (i_1 \cup (i_1 \cdot i_2) \cup \ldots \cup (i_1 \cdot \ldots \cdot i_{n-1})))$, a contradiction.

Hence, since $R_i(w_0) \subseteq R^*_u(w_0) = W$ and $\langle w_0, w_0 \rangle \in R_i$, $\mathcal{M}', w_0 \models \langle i \rangle \top \wedge [i]\phi'$.

(ii) $\rightarrow$ (i). Assume that $\langle i \rangle \top \wedge [i]\phi'$ is $\mathcal{L}^{\mathcal{G}}_m$-satisfiable. So, there is a $\mathcal{L}^{\mathcal{G}}_m$-model $\mathcal{M} = \langle W, R_1, \ldots, R_m, V \rangle$ and $w \in W$ such that $\mathcal{M}, w \models \langle i \rangle \top \wedge [i]\phi'$. So, there is $w_0 \in R_i(w)$ and for all $w' \in R^*_u(w_0)$, $\mathcal{M}, w' \models \phi'$ since $R_i \circ R^*_u \subseteq R_i$. Let $\mathcal{M}' = \langle W', R', V' \rangle$ be the Kripke structure such that $W' \stackrel{\mathrm{def}}{=} R^*_u(w_0)$, R' is the restriction of R_u to W' and V' is the restriction of V to W'. For all $w \in W'$, $\mathcal{M}', w \models \phi$.

We recall that the problem of deciding whether $\mathrm{LAN}(e) \cap \mathrm{LAN}(e') = \emptyset$ is in **P** (see e.g. [HRS76]) where e and e' are regular expressions.

Example 41 Let $\mathcal{G} = \langle \{1\}, \{2, \ldots, m\}, \{1 \rightarrow 1 \cdot 2 \cdot 2 \cdot 3 \cdot 3 \cdot \ldots \cdot m \cdot m\}, 1 \rangle$ be a regular grammar for some $m \geq 3$. $\mathcal{L}^{\mathcal{G}}_m$-satisfiability is **EXPTIME**-complete.

Corollary 42 Let $\mathcal{G}$ be a left linear grammar with a unique production rule of the form $i \rightarrow i \cdot u$ for some $i \in N$ and $u \in \Sigma^+$. Then, $\mathcal{L}^{\mathcal{G}}_m$-satisfiability is **EXPTIME**-hard.

By contrast, for any right linear grammar $\mathcal{G}$ with a unique production rule, $\mathcal{L}^{\mathcal{G}}_m$-satisfiability is in **PSPACE** (see Theorem 37).

8.2 Right linear grammars

Let $\mathcal{G}$ be a right linear grammar such that $card(N \cup \Sigma) \geq 2$. Theorem 38 has a counterpart for right linear grammars.

Theorem 43 Assume that $i \Rightarrow_{\mathcal{G}}^{*} u^{k''} \cdot i$ and $i \Rightarrow_{\mathcal{G}}^{*} u^{k'} \cdot v$ such that

- $i \in N$; $u = i_1 \cdot \ldots \cdot i_n$; $n, k'' \geq 1$;
- $k' \geq 0$; $v = i_1 \cdot \ldots \cdot i_{n'}$; $n' \in \{0, \ldots, n-1\}$ (if $n' = 0$, then $v = \epsilon$);
- there is $j \in \{1, \ldots, n\}$ such that i_j occurs only once in u.

Then, $\mathcal{L}_m^{\mathcal{G}}$-satisfiability is **EXPTIME**-hard.

From Theorem 43, we can reasonably conjecture that **EXPTIME**-hardness is less common for right linear grammar logics than for left linear grammar logics (to be compared with Theorem 38).

Proof Some preliminary remarks are necessary. Let $\mathcal{M} = \langle W, R_1, \ldots, R_m, V \rangle$ be an $\mathcal{L}_m^{\mathcal{G}}$-model. By Theorem 6, $(R_u^{k''})^* \circ R_i \subseteq R_i$ and $(R_u^{k'}) \circ R_v \subseteq R_i$. Consequently, for $\alpha \in \{0, \ldots, k''-1\}$, $R_u^{\alpha} \circ (R_u^{k''})^* \circ R_i \subseteq R_u^{\alpha} \circ R_i$. Hence, $R_u^* \circ R_i \subseteq (\bigcup_{0 \leq \alpha \leq k''-1} R_u^{\alpha}) \circ R_i$. Since $\circ$ is a monotonous operation on relations, we have

$$R_u^* \circ R_u^{k'} \circ R_v \subseteq (\bigcup_{0 \leq \alpha \leq k''-1} R_u^{\alpha}) \circ R_i.$$

Hence,

$$R_u^* \circ R_u^{k'+1} \subseteq (\bigcup_{0 \leq \alpha \leq k''-1} R_u^{\alpha}) \circ R_i \circ R_{i_{n'+1} \cdot \ldots \cdot i_n}.$$

Consequently, $R_u^* \subseteq (\bigcup_{0 \leq \alpha \leq k''-1} R_u^{\alpha}) \circ R_i \circ R_{i_{n'+1} \cdot \ldots \cdot i_n} \cup (\bigcup_{0 \leq \alpha \leq k'} R_u^{\alpha})$. If $i \Rightarrow_{\mathcal{G}}^{*} \epsilon$, then $R_u^* \subseteq (\bigcup_{0 \leq \alpha \leq k''-1} R_u^{\alpha}) \circ R_i$. Let us define a logarithmic space transformation from K-GSAT into $\mathcal{L}_m^{\mathcal{G}}$-satisfiability. Let ϕ be a monomodal formula. Let $g(\phi)$ be the formula

$$\bigwedge_{0 \leq \alpha \leq k''-1} [u^{\alpha} \cdot i \cdot i_{n'+1} \cdot \ldots \cdot i_n] \phi' \wedge \bigwedge_{0 \leq \alpha \leq k'} [u^{\alpha}] \phi'$$

where ϕ' is obtained from the formula ϕ by replacing every occurrence of $\Box$ by $[u]$. If $i \Rightarrow_{\mathcal{G}}^{*} \epsilon$, then $g(\phi)$ can be simplified into $\bigwedge_{0 \leq \alpha \leq k''-1} [u^{\alpha} \cdot i] \phi'$. Let us show that (i) ϕ belongs to K-GSAT iff (ii) $g(\phi)$ is $\mathcal{L}_m^{\mathcal{G}}$-satisfiable. (i) $\rightarrow$ (ii) Assume that ϕ belongs to K-GSAT. Hence, there is a Kripke

structure $\mathcal{M} = \langle W, R, V\rangle$ such that $\mathcal{M} \models \phi$. Without any loss of generality, we can assume that $W = R^*(w_0)$ for some $w_0 \in W$. We define an $\mathcal{L}_m^{\mathcal{G}}$-model $\mathcal{M}' = \langle W', R_1, \ldots, R_m, V'\rangle$ as in the proof of Theorem 38, part (i) $\rightarrow$ (ii). Since for $l \in \Sigma \setminus \{i_j\}$, $R_l = Id_W$ and by assumption i_j occurs only once in u, we have $R_{i_j} = R_u = R$. By Lemma 11, $\mathcal{M}'$ is an $\mathcal{L}_m^{\mathcal{G}}$-model. Since $R_u^*(w_0) = W$, $\mathcal{M}', w_0 \models g(\phi)$.
(ii) $\rightarrow$ (i) Assume that $g(\phi)$ is $\mathcal{L}_m^{\mathcal{G}}$-satisfiable. So, there is a $\mathcal{L}_m^{\mathcal{G}}$-model $\mathcal{M} = \langle W, R_1, \ldots, R_m, V\rangle$ and $w_0 \in W$ such that $\mathcal{M}, w_0 \models g(\phi)$. So for all $w' \in R_u^*(w_0)$, $\mathcal{M}, w' \models \phi'$. Let $\mathcal{M}' = \langle W', R', V'\rangle$ be the Kripke structure such that $W' \stackrel{\text{def}}{=} R_u^*(w_0)$, R' is the restriction of R_u to W' and V' is the restriction of V to W'. For all $w \in W'$, $\mathcal{M}', w \models \phi$.

Example 44 Let $\mathcal{G} = \langle \{1\}, \{2, \ldots, m\}, \{1 \rightarrow \epsilon, 1 \rightarrow (m \cdot m - 1 \cdot \ldots \cdot 2)^m \cdot 1\}, 1\rangle$ be a right linear grammar for some $m \geq 2$. By application of Theorem 43, $\mathcal{L}_m^{\mathcal{G}}$-satisfiability is **EXPTIME**-hard. Moreover, for any regular grammar of the form $\mathcal{G}^+ = \langle \{1\}, \{2, \ldots, m\}, \{1 \rightarrow \epsilon, 1 \rightarrow (m \cdot m - 1 \cdot \ldots \cdot 2)^m \cdot 1\} \cup P^+, 1\rangle$, $\mathcal{L}_m^{\mathcal{G}^+}$-satisfiability is also **EXPTIME**-complete.

Example 45 For $m \geq 2$, $i \geq 1$, let $\mathcal{G}_{m,i}$ be $\langle \{1\}, \{2, \ldots, m\}, \{1 \rightarrow 2^i \cdot 1, 1 \rightarrow \epsilon\}, 1\rangle$. By Theorem 43, $\mathcal{L}_m^{\mathcal{G}_{m,i}}$-satisfiability is **EXPTIME**-hard. For distinct prime numbers i, i', $\mathcal{L}_m^{\mathcal{G}_{m,i}}$-satisfiability and $\mathcal{L}_m^{\mathcal{G}_{m,i'}}$-satisfiability are distinct problems.

Theorem 43 can be easily generalized as follows.

Theorem 46 Assume there are $u_1, \ldots, u_s \in (\Sigma \cup N)^*$ and $u = i_1 \cdot \ldots \cdot i_n \in \Sigma^+$ such that for some $j \in \{1, \ldots, n\}$, i_j occurs only once in u. If $\{u\}^* \subseteq \bigcup_{1 \leq i \leq n} \{u' \in \Sigma^* : u_i \Rightarrow_{\mathcal{G}}^* u'\}$, then $\mathcal{L}_m^{\mathcal{G}}$-satisfiability is **EXPTIME**-hard.

In order to prove Theorem 46, replace in the proof of Theorem 43, the definition of g by $g(\phi) = \bigcup_{1 \leq \alpha \leq n} [u_i]\phi'$ where ϕ' is obtained from ϕ by replacing $\Box$ by $[u]$. Furthermore, although Theorem 46 seems to be quite general, it does not capture all the right linear grammars that generate exponential-time hard logics. For instance, one can show (using some quite ad hoc method) that $\mathcal{L}_3^{\mathcal{G}}$-satisfiability is **EXPTIME**-hard with $\mathcal{G} = \langle \{1\}, \{2, 3\}, \{1 \rightarrow 2 \cdot 2 \cdot 3 \cdot 3 \cdot 1, 1 \rightarrow 2\}, 1\rangle$. $\mathcal{G}$ does not fall into any previous identified cases. Extensions of the **EXPTIME** lower bound to weakly transitive (poly)modal logics (see e.g. [Kra99]) are also expected.

8.3 Decidable criteria for classification

Theorem 38, Theorem 40 and Theorem 43 provide sufficient conditions to guarantee that a given regular grammar logic has an **EXPTIME**-hard satisfiability problem. Those results are partly satisfactory since no equivalence conditions for **EXPTIME**-hardness are provided[1] (this could be part of future work) and we ignore whether the assumptions in the three above mentioned theorems are decidable. For instance, we do not know whether the problem below (related to Theorem 38) is decidable:

- Input: A left linear grammar $\mathcal{G}$.
- Question: Are there a nonterminal symbol i, $k' \geq 1$ and a terminal string u with at least one terminal symbol occurring exactly once such that $i \Rightarrow_{\mathcal{G}}^{*} i \cdot u^{k'}$?

In order to overcome the difficulty of the decidability of the assumptions in Theorems 38, 40 and 43, one can replace in the statements of those theorems the relation $\Rightarrow_{\mathcal{G}}^{*}$ by decidable subrelations. Let g be a computable map from the class REG of regular grammars into the set of natural numbers. For any regular grammar $\mathcal{G}$ we write $\Rightarrow^{\leq g(\mathcal{G})}$ to denote the restriction of $\Rightarrow_{\mathcal{G}}^{*}$ by considering at most $g(\mathcal{G})$ steps for $\Rightarrow_{\mathcal{G}}$ derivations.

Lemma 47 Let $(T_{\mathcal{G}})_{\mathcal{G}\in\mathrm{REG}}$ be a family of relations such that for $\mathcal{G} \in \mathrm{REG}$, $T_{\mathcal{G}} \subseteq \Rightarrow^{\leq g(\mathcal{G})}$. For any regular grammar $\mathcal{G}$, for $u \in (N \cup \Sigma)^*$, $\{v \in (N \cup \Sigma)^* : uT_{\mathcal{G}}v\}$ has at most $(g(\mathcal{G}) \times (card(P) \times |u|)^{g(\mathcal{G})})$ elements.

Proof The elements of $\{v \in (N \cup \Sigma)^* : uT_{\mathcal{G}}v\}$ can be organized as a tree of maximal depth $g(\mathcal{G})$ and of branching factor $card(P) \times |u|$, which roughly provides the announced cardinality bound.

Example 48 Let $\mathcal{G}$ be a regular grammar. A *non-recursive derivation* (see e.g. [Bal98]) is a sequence of the form $u_0 \Rightarrow_{\mathcal{G}} u_1 \Rightarrow_{\mathcal{G}} \ldots \Rightarrow_{\mathcal{G}} u_\alpha$ such that $u_0 \in N$ and every nonterminal symbol different from u_0 occurring in $u_0 \cdot \ldots \cdot u_\alpha$ occurs exactly once and u_0 occurs at most twice. The binary relation $T_{\mathcal{G}}$ subrelation of $\Rightarrow_{\mathcal{G}}^{*}$ is defined as the set of pairs $\langle u_0, u_\alpha \rangle$. Since $\alpha \leq card(N)$, $T_{\mathcal{G}} \subseteq \Rightarrow^{|\mathcal{G}|}$. The non-recursive derivations have been used in [Bal98, BGM98] to show decidability of right linear grammar logics.

[1] Since it is open whether **PSPACE** = **EXPTIME**, the best we could do is to find a partition $\{X_1, X_2\}$ of REG such that for $\mathcal{G} \in X_1$ [resp. $\mathcal{G} \in X_2$], $\mathcal{L}_m^{\mathcal{G}}$-satisfiability is in **PSPACE** [resp. is **EXPTIME**-hard].

Theorem 49 Let g be a computable map $g : \mathrm{REG} \to \mathbb{N}$ and $(T_{\mathcal{G}})_{\mathcal{G}\in\mathrm{REG}}$ be a family of relations such that for $\mathcal{G} \in \mathrm{REG}$, $T_{\mathcal{G}} \subseteq \Rightarrow^{\leq g(\mathcal{G})}$. Then, (I) the statements of Theorems 38, 40 and 43 hold true if we replace $\Rightarrow^*_{\mathcal{G}}$ by $T_{\mathcal{G}}$ and (II) the assumptions on grammars of the new theorems are decidable.

Proof (I) Immediate since $T_{\mathcal{G}} \subseteq \Rightarrow^*_{\mathcal{G}}$. (II) Immediate by Lemma 47.

When $T_{\mathcal{G}}$ is defined as in Example 48, checking whether a grammar satisfies the assumptions of any new theorem in Theorem 49(I) can be done in at most exponential time in $|\mathcal{G}|$.

The class LLIN of left linear grammars generates the same class of formal languages as the class of RLIN of right linear grammars. From the previous results, there is an asymmetry between such classes of grammars as far as **EXPTIME**-hardness of grammar logics is concerned. It is not difficult to show that $\{\mathrm{SF}_i(\mathcal{G}) : i \in \mathcal{G},\ \mathcal{G} \in \mathrm{LLIN}\}$ and $\{\mathrm{SF}_i(\mathcal{G}) : i \in \mathcal{G},\ \mathcal{G} \in \mathrm{RLIN}\}$ are distinct classes of languages which may provide a clue to explain the above mentioned asymmetry. Nevertheless, one might observe that our complexity results do not exhaust all the possibilities for regular grammar logics and therefore, we cannot exclude the possibility that the current asymmetry could be only due to our partial knowledge.

9 Concluding remarks

Let us mention a few open problems that are worth investigating following the results presented in this work.

Classification. Is it the case that any regular grammar logic has a satisfiability problem that is either **EXPTIME**-hard or in **PSPACE**? If the answer is positive, then is it a decidable problem to check whether a regular grammar logic is **EXPTIME**-hard or in **PSPACE**?

First-order fragments. In [GMV99], it is shown that the guarded fragment without equality, with five built-in transitive relations, and two variables is undecidable (see also [Grä99a]). Is there a simple first-order extension of either FO2 or the guarded fragment that is decidable and such that GSP(REG) can be naturally translated into it? We already know that the satisfiability problem of every regular grammar logic can be translated into satisfiability for the guarded fixed point logic μLGF

that is in **EXPTIME** [Grä99b] when the relation symbols have bounded arity. Indeed, PDL can be translated into the modal μ-calculus and the modal μ-calculus can be translated in polynomial-time into μLGF. The design of a simple polynomial-time transformation from GSP(REG) into satisfiability for μLGF is open (for instance μLGF does not allow fixed point predicates in guards). We only know that a polynomial-time many-one reduction exists from the respective complexity characterization of GSP(REG) and μLGF with bounded arity.

Formula size as parameter. Parametrized complexity is a powerful framework to study the complexity of problems where in the inputs, parameters can be distinguished (see e.g. [DF99]). It would be worth investigating the parametric complexity of the parametric version of GSP(REG) where the size of the input formula is the parameter. Where does it belong to the so-called W hierarchy of Downey and Fellows (if it belongs to it)?

Acknowledgments: The remarks and suggestions of the anonymous referees on a previous version of this work were extremely valuable and helpful to improve the quality of this paper.

References

[ABB97] J.M. Autebert, J. Berstel, and L. Boasson. Context-free languages. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, pages 111–174. Springer-Verlag, 1997. Preprint available via http://www-igm.univ-mlv.fr/∼berstel/Recherche.html.

[ABM00] C. Areces, P. Blackburn, and M. Marx. Complexity of hybrid temporal logics. *Journal of the IGPL*, 8(5):653–679, 2000. Available via http://www3.oup.co.uk/igpl/contents/ on WWW.

[ANB98] H. Andreka, I. Nemeti, and J. van Benthem. Modal languages and bounded fragments of predicate logic. *Journal of Philosophical Logic*, 27(3):217–274, 1998.

[Are00] C. Areces. *Logic Engineering: The Case of Description and Hybrid Logics*. PhD thesis, University of Amsterdam, 2000. Available via http://turing.wins.uva.nl/∼carlos/ on WWW.

[Bal98] M. Baldoni. *Normal Multimodal Logics: Automated Deduction and Logic Programming*. PhD thesis, Università degli Studi di Torino, 1998. Available via http://www.di.unito.it/∼baldoni on WWW.

[Ben76] J. van Benthem. Modal reduction principles. *The Journal of Symbolic Logic*, 2:301–312, 1976.

[Ben84] J. van Benthem. Correspondence Theory. In D. Gabbay and F. Günthner, editors, *Handbook of Philosophical Logic, Volume II*, pages 167–247. Reidel, Dordrecht, 1984.

[Ben98a] J. van Benthem. Modal Logic in Two Gestalts. Technical Report ML-98-12, ILLC, Amsterdam, 1998. Available via http://turing.wins.uva.nl/~johan/papers90.html on WWW.

[Ben98b] J. van Benthem. Program constructions that are safe for bisimulation. *Studia Logica*, 60:311–330, 1998.

[BGM98] M. Baldoni, L. Giordano, and A. Martelli. A tableau calculus for multimodal logics and some (un)decidability results. In H. de Swart, editor, *TABLEAUX-8*, pages 44–59. LNAI 1397, Springer-Verlag, 1998. Available via http://www.di.unito.it/~baldoni on WWW.

[BM75] J. de Bakker and G. Meertens. On the completeness of the inductive assertion method. *Journal of Computer and System Sciences*, 11:323–357, 1975.

[BRV01] P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic*. Cambridge University Press, 2001. Preprint available via http://turing.wins.uva.nl/~mdr/ on WWW.

[Cat89] L. Catach. *Les logiques multimodales*. PhD thesis, Université Pierre et Marie Curie (Paris 6), 1989.

[CL94] C. Chen and I. Lin. The complexity of propositional modal theories and the complexity of consistency of propositional modal theories. In A. Nerode and Yu. V. Matiyasevich, editors, *LFCS-3, St. Petersburg*, pages 69–80. Springer-Verlag, LNCS 813, 1994.

[CS94] A. Chagrov and V. Shehtman. Algorithmic aspects of propositional tense logics. In L. Pacholski and J. Tiuryn, editors, *CSL-8, Kazinierz, Poland*, pages 442–455. LNCS 933, Springer Verlag, 1994.

[Dem01] S. Demri. Modal logics with weak forms of recursion: PSPACE specimens. In M. de Rijke, H. Wansing, F. Wolter, and M. Zakharyaschev, editors, *Selected papers from 3rd Workshop on Advances in Modal Logics (AIML'2000), Leipzig, Germany, Oct. 2000*. CSLI, 2001. To appear. Preliminary version available via http://www.lsv.ens-cachan.fr/~demri/ on WWW.

[DF99] R. Downey and M. Fellows. *Parameterized complexity*. Springer-Verlag, 1999.

[DLNN97] F. Donini, M. Lenzerini, D. Nardi, and W. Nutt. The complexity of concept languages. *Information and Computation*, 134:1–58, 1997.

[EHR83] A. Ehrenfeucht, D. Haussler, and G. Rozenberg. On regularity of context-free languages. *Theoretical Computer Science*, 27:311–332, 1983.

[EJ99] A. Emerson and C. Jutla. The complexity of tree automata and logics of programs. *SIAM Journal of Computing*, 29:132–158, 1999. Journal version of the FOCS'88 paper.

[EZ76] A. Ehrenfeucht and P. Zeiger. Complexity measures for regular expressions. *Journal of Computer and System Sciences*, 12:134–146, 1976.

[FdCH95] L. Fariñas del Cerro and A. Herzig. Modal deduction with applications in epistemic and temporal logics. In D. Gabbay, C. J. Hogger, and J. A. Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming*, pages 499–593, 1995.

[FdCP88] L. Fariñas del Cerro and M. Penttonen. Grammar logics. *Logique et Analyse*, 121-122:123–134, 1988.

[FL79] M. Fischer and R. Ladner. Propositional dynamic logic of regular programs. *Journal of Computer and System Sciences*, 18:194–211, 1979.

[Gab81] D. Gabbay. Expressive functional completeness in tense logic. In U. Mönnich, editor, *Aspects of Philosophical Logic*, pages 91–117. Reidel, Dordrecht, 1981.

[Gas94] O. Gasquet. *Déduction automatique en logique multi-modale par traduction.* PhD thesis, Université Paul Sabatier, Toulouse, 1994. Available via http://www.irit.fr/ on WWW.

[Gia95] G. de Giacomo. *Decidability of Class-Based Knowledge Representation Formalisms.* PhD thesis, Universita Degli Studi Di Roma 'La Sapienza', 1995. Available via http://www.dis.uniroma1.it/∼degiacom/publications.html on WWW.

[GL94] G. de Giacomo and M. Lenzerini. Boosting the correspondence between description logics and propositional dynamic logics. In *AAAI'94*, pages 205–212. AAAI Press/The MIT Press, 1994.

[GMV99] H. Ganzinger, C. Meyer, and M. Veanes. The two-variable guarded fragment with transitive relations (extended abstract). In *LICS'99*, pages 24–34, 1999. Available via http://www.mpi-sb.mpg.de/∼hg on WWW.

[Gol81] L. Goldschlager. ϵ-productions in context-free grammars. *Acta Informatica*, 16(3):303–308, 1981.

[GOR97] E. Grädel, M. Otto, and E. Rosen. Undecidability results on two-variable logics. In *STACS-14*, pages 249–260. LNCS 1200, Springer-Verlag, 1997. Available via http://www-mgi.informatik.rwth.aachen.de/Publications on WWW.

[Grä99a] E. Grädel. On the restraining power of guards. *The Journal of Symbolic Logic*, 64(4):1719–1742, 1999.

[Grä99b] E. Grädel. Why are modal logics so robustly decidable? *Bulletin of the EATCS*, 68:90–103, 1999.

[GW99] E. Grädel and I. Walukiewicz. Guarded fixed point logic. In *LICS'99*, pages 45–54, 1999.

[Har58] R. Harrop. On the existence of finite models and decision procedures for propositional calculi. *Proceedings of the Cambridge Philosophical Society*, 54:1–13, 1958.

[Har84] D. Harel. Dynamic logic. In D. Gabbay and F. Günthner, editors, *Handbook of Philosophical Logic, Volume II*, pages 497–604. Reidel, Dordrecht, 1984.

[Hem94] E. Hemaspaandra. Complexity transfer for modal logic (extended abstract). In *LICS-9*, pages 164–173, 1994.

[Hem96] E. Hemaspaandra. The price of universality. *Notre Dame Journal of Formal Logic*, 37(2):173–203, 1996.

[HKT00] D. Harel, D. Kozen, and J. Tiuryn. *Dynamic Logic*. MIT Press, 2000.

[HM92] J. Halpern and Y. Moses. A guide to completeness and complexity for modal logics of knowledge and belief. *Artificial Intelligence*, 54:319–379, 1992.

[HM97] W. van der Hoek and J.-J. Meyer. A complete epistemic logic for multiple agents - combining distributed and common knowledge. In P. Mongin, M. Bacharach, L. Gerard-Valet, and H. Shin, editors, *Epistemic Logic and the Theory of Games and Decisions*, pages 35–68, 1997.

[HM00] V. Haarslev and R. Möller. Expressive ABox reasoning with number restrictions, role hierarchies, and transitively closed roles. In F. Giunchiglia and B. Selman, editors, *7th International Conference on Principles of Knowledge Representation and Reasoning*, pages 273–284. Morgan Kaufmann, 2000. Preprint avail-

able via http://kogs-www.informatik.uni-hamburg.de/~haarslev/ on WWW.

[HPS83] D. Harel, A. Pnueli, and J. Stavi. Propositional dynamic logic of nonregular programs. *Journal of Computer and System Sciences*, 26:222–243, 1983.

[HR93] D. Harel and D. Raz. Deciding properties of nonregular programs. *SIAM Journal on Computing*, 22:857–874, 1993.

[HRS76] H. Hunt, D. Rosenkrantz, and Th. Szymanski. On the equivalence, containment, and covering problems for the regular and context-free languages. *Journal of Computer and System Sciences*, 12:222–268, 1976.

[HS99] I. Horrocks and U. Sattler. A description logic with transitive and inverse roles and role hierarchies. *Journal of Logic and Computation*, 9(3):385–410, 1999.

[HST00] I. Horrocks, U. Sattler, and S. Tobies. Practical reasoning for very expressive description logics. *Logic Journal of IGPL*, 8(3):239–263, 2000. Available via http://www3.oup.co.uk/igpl/contents/ on WWW.

[HU79] J. Hopcroft and J. Ullman. *Introduction to automata theory, languages, and computation.* Addison-Wesley Publishing Company, Reading, M.A., 1979.

[Hus99] U. Hustadt. *Resolution-Based Decision Procedures for Subclasses of First-Order Logic.* PhD thesis, Fakultät der Universität des Saarlandes, 1999.

[JL76] N. Jones and W. Laaser. Complete problems for deterministic polynomial-time. *Theoretical Computer Science*, 3(1):105–117, 1976.

[Kle56] S. Kleene. Representation of events in nerve nets. In C. Shannon and J. McCarthy, editors, *Automata studies*, pages 3–40. Princeton University Press, Princeton, NJ, 1956.

[Koz83] D. Kozen. Results on the propositional μ-calculus. *Theoretical Computer Science*, 27:333–354, 1983.

[Kra96] M. Kracht. Power and weakness of the modal display calculus. In H. Wansing, editor, *Proof theory of modal logic*, pages 93–121. Kluwer, 1996. Available via http://www.math.fu-berlin.de/user/kracht/ on WWW.

[Kra99] M. Kracht. *Tools and Techniques in Modal Logic.* Elsevier, 1999.

[Lad77] R. Ladner. The computational complexity of provability in systems of modal propositional logic. *SIAM Journal of Computing*, 6(3):467–480, 1977.

[Lut99] C. Lutz. Complexity of terminological reasoning revisited. In *LPAR'99*. LNAI 1795, 1999. Available via http://www-lti.informatik.rwth-aachen.de/~clu/papers/index.html on WWW.

[Mas00] F. Massacci. Single steps tableaux for modal logics. *Journal of Automated Reasoning*, 24(3):319–364, 2000. Available via http://www.dis.uniroma1.it/~massacci/papers/ on WWW.

[Min88] G. Mints. Gentzen-type and resolution rules part I: propositional logic. In P. Martin-Löf and Grigori Mints, editors, *International Conference on Computer Logic, Tallinn*, pages 198–231. Springer Verlag, LNCS 417, 1988.

[MS97] A. Mateescu and A. Salomaa. Formal languages: an introduction and a synopsis. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages - Volume 1: Word, Language and Grammar*, pages 1–40. Springer, 1997.

[Niv98] H. de Nivelle. A resolution decision procedure for the guarded fragment. In C. Kirchner and H. Kirchner, editors, *CADE-15, Lindau, Germany*, pages 191–204. LNAI 1421, Springer-Verlag, 1998. Available via http://www.mpi-sb.mpg.de/~nivelle/ on WWW.

[Pap94] Ch. Papadimitriou. *Computational Complexity*. Addison-Wesley Publishing Company, 1994.

[Per90] D. Perrin. Finite automata. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science, Volume B, Formal models and semantics*, pages 1–57. Elsevier, 1990.

[Pra79] V. Pratt. Models of program logics. In *Proceedings of the 20th IEEE Symposium on Foundations of Computer Science*, pages 115–122, 1979.

[Pra81] V. Pratt. Using graphs to understand PDL. In D. Kozen, editor, *Workshop on Logics of Programs*, pages 387–396. LNCS 131, Springer-Verlag, 1981.

[PT91] S. Passy and T. Tinchev. An essay in combinatory dynamic logic. *Information and Computation*, 93:263–332, 1991.

[RS94] G. Rozenberg and A. Salomaa. *Cornerstones of Undecidability*. Prentice Hall, 1994.

[Sah75] H. Sahlqvist. Completeness and correspondence in the first and second order semantics for modal logics. In S. Kanger, editor,

3rd Scandinavian Logic Symposium, Uppsala, Sweden, 1973, pages 110–143. North Holland, 1975.

[Sat96] U. Sattler. A concept language extended with different kinds of transitive roles. In *20. Deutsche Jahrestagung für Künstliche Intelligenz*. LNM 1137, Springer, 1996.

[Sch62] M. Schützenberger. On a theorem of R. Jungen. *Proc. Amer. Math. Soc.*, 13:885–889, 1962.

[Sch91] K. Schild. A correspondence theory for terminological logics: preliminary report. In *IJCAI-12*, pages 466–471, 1991.

[Sch97] R. Schmidt. *Optimised Modal Translation and Resolution*. PhD thesis, Fakultät der Universität des Saarlandes, 1997.

[Spa93] E. Spaan. *Complexity of Modal Logics*. PhD thesis, ILLC, Amsterdam University, 1993.

[SS89] M. Schmidt-Schauss. Subsumption in KL-ONE is undecidable. In R. Brachman, H. Levesque, and R. Reiter, editors, *KR'89*, pages 421–431. Morgan Kaufmann, 1989.

[SSS91] M. Schmidt-Schauss and G. Smolka. Attribute concept descriptions with complements. *Artificial Intelligence*, 48(1):1–26, 1991.

[TK91] M. Tiomkin and M. Kaminsky. Nonmonotonic default modal logics. *Journal of the Association for Computing Machinery*, 38:963–984, 1991.

[Tuo90] H. Tuominen. Dynamic logic as a uniform framework for theorem proving in intensional logic. In M. E. Stickel, editor, *CADE-10*, pages 514–527. LNCS 449, Springer-Verlag, 1990.

[Var97] M. Vardi. Why is modal logic so robustly decidable? In *DIMACS Series in Discrete Mathematics and Theoretical Computer Science 31, American Mathematical Society*, pages 149–183, 1997. Available via http://www.cs.rice.edu/∼vardi/ on WWW.

[VW86] M. Vardi and P. Wolper. Automata-theoretic techniques for modal logics of programs. *Journal of Computer and System Sciences*, 32:183–221, 1986.

[VW94] M. Vardi and P. Wolper. Reasoning about infinite computations. *Information and Computation*, 115:1–37, 1994. Journal version of the FOCS'83 paper.

[WHM00] M. Wessel, V. Haarslev, and R. Möller. $\mathrm{ALC}_{\mathrm{RA}}$ - ALC with role axioms. In F. Baader and U. Sattler, editors, *International Workshop in Description Logics*, pages 267–276, 2000. Available via

http://sunsite.informatik.rwth.aachen.de/Publications/CEUR-WS/Vol-33 on WWW.

[Wol83] P. Wolper. Temporal logic can be more expressive. *Information and Control*, 56:72–99, 1983.

[Wol99] F. Wolter. *The decision problem for combined (modal) logics*. Kluwer Academic Publishers, Dordrecht, 1999. to appear.

[Zak97] M. Zakharyaschev. Canonical formulas for K4. part III: the finite model property. *The Journal of Symbolic Logic*, 62(3):950–975, 1997.