Beyond Regularity for Presburger Modal Logics

Facundo Carreiro, Stéphane Demri

Beyond Regularity for Presburger Modal Logics

Facundo Carreiro [1]

ILLC, University of Amsterdam, the Netherlands

Stéphane Demri

LSV, ENS Cachan, CNRS, INRIA, France

Abstract

Satisfiability problem for modal logic K with quantifier-free Presburger and regularity constraints (EML) is known to be PSPACE-complete. In this paper, we consider its extension with nonregular constraints, and more specifically those expressed by visibly pushdown languages (VPL). This class of languages behaves nicely, in particular when combined with Propositional Dynamic Logic (PDL). By extending EML, we show that decidability is preserved if we allow at most one positive VPL-constraint at each modal depth. However, the presence of two VPL-contraints or the presence of a negative occurrence of a single VPL-constraint leads to undecidability. These results contrast with the decidability of PDL augmented with VPL-constraints.

Keywords: Presburger constraint, context-free constraint, decidability

1 Introduction

Presburger modal logics. Graded modal logics are extensions of modal logic K in which the modality $\Diamond$ (see e.g. [5]) is replaced by $\Diamond_{\geqslant n}$; the formula $\Diamond_{\geqslant n}\, p$ states that there are at least n successor worlds satisfying the proposition p, early works about such counting operators can be found in [15,4,10]. In [30], the minimal graded modal logic, counterpart of the modal logic K, is shown decidable in PSPACE. Recent complexity results can be found in [20], see also enriched logics with graded modalities in [21,6]. Nevertheless, richer arithmetical constraints about successor worlds are conceivable. For instance, in the majority logic [25], one can express that more than half of the successors satisfy a given formula. All of the above constraints, and many more, are expressible in Presburger arithmetic (PrA), the decidable first-order theory of natural numbers with addition. In the sequel, Presburger modal logics refer to logics admitting arithmetical constraints from PrA. Not only arithmetical constraints from graded modal logics have been considered in several classes of

[1] F. Carreiro has been supported by INRIA at LSV, ENS Cachan and by CONICET Argentina at the Department of Computer Science of the University of Buenos Aires.

non-classical logics, including epistemic logics [31] and description logics [18,9,3] but also PSPACE-complete logics with richer arithmetical constraints have been designed [27,13,11]. If the PSPACE upper bound is given up, modal-like logics with more expressive Presburger constraints on the number of children can be found in [21,34,28].

Assuming that the children of a node are ordered, it is possible to enrich Presburger modal logics with regularity constraints related to the ordering of siblings. This is particularly meaningful to design logical formalisms to query XML documents viewed as finite ordered unranked labeled trees; such logical and automata-based formalisms can be found, e.g., in [34,28]. For instance, a logic with fixpoint operators, arithmetical and regularity constraints is introduced in [28] and shown decidable with an exponential-time complexity, which improves results for description logics with qualified number restrictions [9]. Similarly, the main goal of [13] was to introduce a Presburger modal logic EML with regularity constraints as in the logical formalisms from [33,34,28] with a satisfiability problem that could still be solved in polynomial space.

Our motivations: beyond regularity. The PSPACE upper bound established in [13] for the satisfiability problem for EML is based on a Ladner-like algorithm [22]. Moreover it takes advantage of the properties that the commutative images of context-free languages (CFLs) are effectively semilinear [26] and satisfiability for existential Presburger formulae can be solved in NP thanks to the existence of small solutions [7]. In this paper, our goal is to revisit decidability and complexity results for Presburger modal logics when context-free constraints are considered instead of regular ones. Usefulness of such contraints is best witnessed by their use to define Document Type Definitions (DTDs) for XML documents or to express nonregular programs as done in [23], but the paper focuses on decidability issues. However, it is not reasonable to expect that adding all context-free languages (CFLs) would preserve decidability. We are aware of several situations in which such an extension does not preserve decidability. For instance, PDL with regular programs augmented with the context-free program $\{\mathtt{a}^i\mathtt{b}\mathtt{a}^i : i \geqslant 0\}$ is known to be undecidable [17]. On the other hand, PDL augmented with context-free programs definable by *semi-simple minded pushdown automata* is decidable [16]. In such pushdown automata, the input symbol determines the stack operation to be performed, the next control state and the symbol to be pushed in case of a push operation. The ultimate decidability result has been obtained for PDL augmented by programs definable by visibly pushdown automata (VPA) [23]. These automata are introduced in [1] (see also the equivalent class of *input-driven pushdown automata* in [24]) and are motivated by verification problems for recursive state machines. They generalize the semi-simple-minded pushdown automata since we only require that the input symbol determines the stack operation.

The initial motivation for this work is to understand the decidability status of EML augmented with context-free constraints from VPAs (*VPL-constraints*). Also, the decidability proof for EML takes advantage of the Boolean closure of

regular languages, which is also the case for visibly pushdown languages [1] (complementation can be performed in exponential time and causes -only- an exponential blow-up [1]). Moreover, by [32], from a VPA, one can compute in linear time a simple Presburger formula whose solutions are the Parikh image of its language. This may allow us to extend [13]. Hence, boosted by the decidability and complexity results from [23] for PDL with VPA and by the nice properties of VPLs, we aim at understanding how much of the power of context-free constraints can be added to EML while still preserving decidability.

Our contributions. We can distinguish three types of contributions: showing that different extensions of EML have identical expressive power, proving undecidability results when context-free constraints are present in formulae and establishing decidability for the satisfiability problem of EML extended with restricted context-free constraints.

- EML extended with VPL-constraints such that at each depth, there are at most i such constraints (written $\mathsf{EML}_i(\mathrm{VPL})$) is expressively equivalent to EML extended with CFL-constraints such that at each depth, there are at most i such constraints (written $\mathsf{EML}_i(\mathrm{CFL})$).
- We show that a strict fragment of $\mathsf{EML}_2(\mathrm{CFL})$, with only positive CFL-constraints, and a strict fragment of $\mathsf{EML}_1(\mathrm{CFL})$, with a single CFL-constraint occurring negatively, have undecidable satisfiability problems.
- Last, but not least, we show that $\mathsf{EML}_1^+(\mathrm{CFL})$, a fragment of $\mathsf{EML}_1(\mathrm{CFL})$ but a substantial extension of EML, has a decidable satisfiability problem by extending the proof technique from [13]. Herein, $\mathsf{EML}_1^+(\mathrm{CFL})$ denotes the fragment of $\mathsf{EML}_1(\mathrm{CFL})$ in which at each depth, there is at most one context-free constraint and it occurs positively plus additional more technical conditions. Positive occurrences of constraints shall be defined in Section 2 and this notion slightly differs from the one that only counts the number of negations. This is the best we can hope for in view of the previous results.

It was unexpected that $\mathsf{EML}_2(\mathrm{VPL})$ is already undecidable. Decidability can be regained if only one VPL-constraint occurs positively at each depth. If only one VPL-constraint may occur negatively at each depth, then undecidability is back. This contrasts with the decidability of PDL extended with VPAs [23] or with the undecidability of LTL extended with a fixed VPA [12]. Since modal logic K is a fragment of PDL, this seems to contradict the undecidability of $\mathsf{EML}_2(\mathrm{CFL})$. However, language-based constraints on PDL are related to paths whereas in EML it is related to the ordering of successor worlds. Moreover, a CFL-constraint $\mathrm{L}(p_1, \ldots, p_n)$ on the children can be turned into an arithmetical constraint by using Parikh's Theorem [26]. In a sense, CFL-constraints on the successor worlds can be handled by arithmetical constraints. Again, this seems to contradict the undecidability of $\mathsf{EML}(\mathrm{CFL})$ fragments; this contradiction vanishes if we recall that the Parikh image of $\overline{\mathrm{L}}$ is not equal to the complement of the Parikh image of L and the Parikh image of the intersection of two CFLs is not equal to the intersection of the Parikh images of the languages. Hence,

only isolated CFL-constraints (without other language-based constraints) can be safely replaced by arithmetical constraints.

2 Preliminaries

In this section, we recall the extended modal logic EML and we present extensions based on CFLs. In particular, we shall recall what VPLs are.

2.1 Extended Modal Logic EML

The logics studied in this paper extend the logic EML that has Presburger and regularity constraints. Given a countably infinite set $\mathrm{AT} = \{p_1, p_2, \dots\}$ of propositional variables, the set of terms and EML formulae is inductively defined as follows

$$t ::= a \times \sharp\varphi \mid t + a \times \sharp\varphi$$

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid t > b \mid t \equiv_k c \mid \mathrm{L}(\varphi_1, \dots, \varphi_n)$$

where

- $b, c \in \mathbb{N}$, $k \in \mathbb{N} \setminus \{0, 1\}$, $a \in \mathbb{Z} \setminus \{0\}$.
- L is any regular language specified by a finite-state automaton.

A model for EML is a structure $\langle T, R, (<_s)_{s\in T}, l\rangle$ where T is the (possibly infinite) set of nodes and $R \subseteq T^2$ such that the set $\{t \in T : \langle s, t\rangle \in R\}$ is finite for each $s \in T$. Each relation $<_s$ is a total ordering on the R-succesors of s and $l : T \to 2^{\mathrm{AT}}$ is the valuation function. We write $R(s) = s_1 < \cdots < s_m$ if the children of s are $s_1, \dots, s_m$ and they are ordered this way. The satisfaction relation is inductively defined as follows (we omit Boolean clauses):

- $\mathcal{M}, s \models p$ iff $p \in l(s)$,
- $\mathcal{M}, s \models \sum_i a_i \sharp\varphi_i \equiv_k c$ iff there is $n \in \mathbb{N}$ such that $\sum_i a_i R^{\sharp}_{\varphi_i}(s) = nk + c$ with $R_{\varphi_i} = \{\langle s', s''\rangle \in T^2 : \langle s', s''\rangle \in R, \text{ and } \mathcal{M}, s'' \models \varphi_i\}$ and $R^{\sharp}_{\varphi_i}(s) = \mathrm{card}(R_{\varphi_i}(s))$,
- $\mathcal{M}, s \models \sum_i a_i \sharp\varphi_i > b$ iff $\sum_i a_i R^{\sharp}_{\varphi_i}(s) > b$,
- $\mathcal{M}, s \models \mathrm{L}(\varphi_1, \dots, \varphi_n)$ iff the finite sequence of children of the node s induces a finite pattern from L. More precisely, iff there is $\mathsf{a}_{i_1} \cdots \mathsf{a}_{i_m} \in \mathrm{L}$ such that given the children of s, $R(s) = s_1 < \cdots < s_m$, for every $j \in [1, m]$, we have $\mathcal{M}, s_j \models \varphi_{i_j}$ (L is any regular language).

A formula φ is *satisfiable* iff there is a model $\mathcal{M}$ and a node s such that $\mathcal{M}, s \models \varphi$. The satisfiability problem is defined accordingly. Observe that although arithmetical constraints are only allowed to use $>$, the operators $\{=, <, \leqslant\}$ can be easily defined using Boolean combinations and constants (no exponential blow-up occurs if renaming is used). Hence, we will use them as abbreviations.

Modalities from K and graded modal logic can also be defined in EML. We use the abbreviation $\Box\varphi \stackrel{\text{def}}{=} \neg(\sharp\neg\varphi > 0)$. Despite the fact that EML extends graded modal logics with richer arithmetical constraints but also with regularity constraints, the satisfiability problem for EML is PSPACE-complete [13].

Now, let us recall what formula trees are. They will be used to define the different fragments that we will be working with. A *finite tree* $\mathcal{T}$ is a finite subset of $(\mathbb{N} \setminus \{0\})^*$ such that $\varepsilon \in \mathcal{T}$ and if $s{\cdot}i \in \mathcal{T}$, then $s \in \mathcal{T}$ and $s{\cdot}j \in \mathcal{T}$ for $j \in [1, i-1]$. The nodes of $\mathcal{T}$ are its elements. The root of $\mathcal{T}$ is the empty word ε. All notions such as parent, child, subtree and leaf, have their standard meanings. A *formula tree* is a labelled tree $\langle \mathcal{T}, \ell \rangle$ representing a formula φ (the label set being a set of subformulae), i.e.

- $\ell(\varepsilon) = \varphi$. If $\ell(s) = \psi_1 \oplus \psi_2$ with $\oplus \in \{\vee, \wedge\}$, then s has two children, $\ell(s{\cdot}1) = \psi_1$ and $\ell(s{\cdot}2) = \psi_2$ ($\neg$ has a similar clause).
- If $\ell(s) = \mathrm{L}(\varphi_1, \ldots, \varphi_n)$, then s has n children and for all $j \in [1, n]$, we have $\ell(s{\cdot}j) = \varphi_j$ (formulae $\sum_i a_i \sharp \varphi_i > b$ and $\sum_i a_i \sharp \varphi_i \equiv_k c$ have similar clauses).

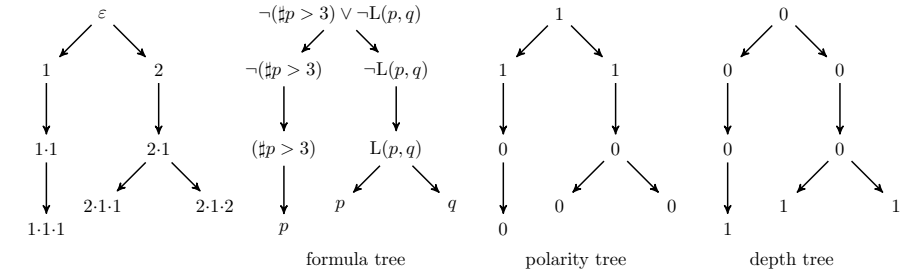
EML formulae are encoded as formula trees and occurrences of ψ in φ correspond to nodes of the formula tree for φ that are labelled by ψ. Given a formula tree $\langle \mathcal{T}, \ell \rangle$, we define its *polarity tree* $\langle \mathcal{T}, \ell_{\mathrm{POL}} \rangle$ as follows:

- $\ell_{\mathrm{POL}}(\varepsilon) = 1$. If $\ell(s) = \neg\psi$, then $\ell_{\mathrm{POL}}(s{\cdot}1) = (1 - \ell_{\mathrm{POL}}(s))$.
- If $\ell(s) = \psi_1 \oplus \psi_2$ with $\oplus \in \{\vee, \wedge\}$, then $\ell_{\mathrm{POL}}(s{\cdot}1) = \ell_{\mathrm{POL}}(s{\cdot}2) = \ell_{\mathrm{POL}}(s)$.
- If $\ell(s) = \sum_i a_i \sharp \varphi_i > b$ or $\ell(s) = \sum_i a_i \sharp \varphi_i \equiv_k c$ or $\ell(s) = \mathrm{L}(\varphi_1, \ldots, \varphi_n)$, then for all j, we have $\ell_{\mathrm{POL}}(s{\cdot}j) = \ell_{\mathrm{POL}}(s)$.

Given a formula tree $\langle \mathcal{T}, \ell \rangle$, we define its *depth tree* $\langle \mathcal{T}, \ell_{\mathrm{DEP}} \rangle$ as follows:

- $\ell_{\mathrm{DEP}}(\varepsilon) = 0$. If $\ell(s) = \neg\psi$, then $\ell_{\mathrm{DEP}}(s{\cdot}1) = \ell_{\mathrm{DEP}}(s)$.
- If $\ell(s) = \psi_1 \oplus \psi_2$ with $\oplus \in \{\vee, \wedge\}$, then $\ell_{\mathrm{DEP}}(s{\cdot}1) = \ell_{\mathrm{DEP}}(s{\cdot}2) = \ell_{\mathrm{DEP}}(s)$.
- If $\ell(s) = \sum_i a_i \sharp \varphi_i \equiv_k c$ or $\ell(s) = \mathrm{L}(\varphi_1, \ldots, \varphi_n)$ or $\ell(s) = \sum_i a_i \sharp \varphi_i > b$ then for all j, we have $\ell_{\mathrm{DEP}}(s{\cdot}j) = 1 + \ell_{\mathrm{DEP}}(s)$.

Intuitively, this last definition assigns the modal-arithmetical depth to the subformula in each node. By way of example, we present the formula tree for $\neg(\sharp p > 3) \vee \neg\mathrm{L}(p, q)$ together with its polarity tree and its depth tree.



2.2 Extensions with CFL-constraints

Let us briefly fix some notations about pushdown automata. A *pushdown automaton* (denoted by PDA) M is a tuple $\langle Q, \Sigma, \Gamma, \delta, q_0, \bot, F \rangle$ where Q is a finite set of states, Σ (resp. Γ) is the finite input (resp. stack) alphabet, δ is a mapping of $Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma$ into finite subsets of $Q \times \Gamma^*$, $q_0 \in Q$ is the

initial state, $\bot$ is the initial stack symbol, $F \subseteq Q$ is the set of accepting states. A configuration $\langle q, \alpha, \gamma\rangle$ is an element of $Q \times \Sigma^* \times \Gamma^*$. Initial configurations are of the form $\langle q_0, \varepsilon, \bot\rangle$. The one-step relation $\vdash$ (with respect to M) between configurations is defined as follows: $\langle q, \alpha, \mathtt{b}\gamma\rangle \vdash \langle q', \alpha\mathtt{a}, \gamma'\gamma\rangle$ whenever there is $\langle q', \gamma'\rangle \in \delta(q, \mathtt{a}, \mathtt{b})$. As usual, $\vdash^*$ is the reflexive and transitive closure of $\vdash$. We write $\mathcal{L}(M)$ to denote the language accepted by the pushdown automaton M when acceptance is by final states, i.e. $\mathcal{L}(M) = \{\alpha \in \Sigma^* : \langle q_0, \varepsilon, \bot\rangle \vdash^* \langle q, \alpha, \gamma\rangle$ with $q \in F\}$.

A *pushdown alphabet* Σ is an alphabet equipped with a partition $\langle \Sigma_c, \Sigma_i, \Sigma_r\rangle$ such that the letters in Σ_c are *calls* (corresponding to push actions on the stack), the letters in Σ_i are *internal actions* (no action on the stack) and the letters in Σ_r are *returns* (corresponding to pop actions on the stack). A *visibly pushdown automaton* (VPA) over Σ is defined as a PDA except that the input letters in Σ determine which actions are performing on the stack according to the partition $\langle \Sigma_c, \Sigma_i, \Sigma_r\rangle$. More precisely, a VPA M over a pushdown alphabet $\langle \Sigma_c, \Sigma_i, \Sigma_r\rangle$ is a PDA without ε-transitions and such that for all states q, input symbols $\mathtt{a} \in \Sigma$ and stack symbols $\mathtt{b} \in \Gamma$,

- $\mathtt{a} \in \Sigma_c$ and $\langle q', \gamma\rangle \in \delta(q, \mathtt{a}, \mathtt{b})$ imply $\gamma = \mathtt{b}'\mathtt{b}$ for some $\mathtt{b}' \in \Gamma$ (call),
- $\mathtt{a} \in \Sigma_i$ and $\langle q', \gamma\rangle \in \delta(q, \mathtt{a}, \mathtt{b})$ imply $\gamma = \mathtt{b}'$ (internal action),
- $\mathtt{a} \in \Sigma_r$ and $\langle q', \gamma\rangle \in \delta(q, \mathtt{a}, \mathtt{b})$ imply $\gamma = \varepsilon$ (return).

A *visibly pushdown language* (VPL) is a language accepted by a VPA. By [1], given two VPLs L_1 and L_2 over the same pushdown alphabet Σ, $\mathrm{L}_1 \cup \mathrm{L}_2$, $\mathrm{L}_1 \cap \mathrm{L}_2$ and $\Sigma^* \setminus \mathrm{L}_1$ are also VPLs and one can effectively compute the corresponding VPA.

Given a class $\mathcal{C}$ of languages (made of finite words from finite alphabets), we define the modal logic $\mathsf{EML}(\mathcal{C})$ interpreted over finitely-branching Kripke structures such that $\mathsf{EML}(\mathcal{C})$ is an extension of EML by allowing formula $\mathrm{L}(\varphi_1, \ldots, \varphi_n)$ with $\mathrm{L} \in (\mathcal{C} \cup \mathrm{REG})$ where REG is the class of regular languages. In this paper, $\mathcal{C}$ is either the class of context-free languages (denoted by CFL) or the class of visibly pushdown languages (denoted by VPL). Of course, elements of $\mathcal{C}$ need to be represented finitely; to do so, when $\mathrm{L} \in \mathrm{CFL}$, L is specified either by a PDA or by a context-free grammar (CFG). Moreover, when $\mathrm{L} \in \mathrm{VPL}$, L is specified by a VPA. Obviously, $\mathsf{EML} = \mathsf{EML}(\mathrm{REG})$. Let $\mathsf{EML}_i(\mathcal{C})$ be the set of formulae φ in $\mathsf{EML}(\mathcal{C})$ such that for any depth j in its depth tree, there are at most i nodes labelled by formulae of the form $\mathrm{L}(\cdots)$ with $\mathrm{L} \in \mathcal{C}$. For example, $\mathrm{L}_1(p, q) \wedge \mathrm{L}_2(p, q) \wedge \neg \mathrm{L}_3(p, q) \in \mathsf{EML}_1(\mathrm{CFL})$ with $\mathrm{L}_1, \mathrm{L}_2 \in \mathrm{REG}$ and $\mathrm{L}_3 \in \mathrm{CFL}$.

2.3 Properties expressible in $\mathsf{EML}(\mathrm{CFL})$

It is obvious that $\mathsf{EML}(\mathrm{CFL})$ is at least as expressive as EML. However, the fact that there are context-free languages that are not regular is not sufficient to conclude that $\mathsf{EML}(\mathrm{CFL})$ is strictly more expressive than EML since the languages are embedded in a logical context. In fact, in Section 3, we show that $\mathsf{EML}(\mathrm{VPL})$ is as expressive as $\mathsf{EML}(\mathrm{CFL})$ even though VPL is strictly

included in CFL. In the paragraphs below, we explore the relationships between properties expressible in EML and those expressible in EML(CFL). Towards the end of the section, we finally show that EML(CFL) is strictly more expressive.

Let us express in EML(CFL) that there are no children of distance exactly 2^{N+1} such that the first child satisfies p and the second one satisfies q. This property can be instrumental for reducing tiling problems and, with it, establish complexity lower bounds. Let L be the language defined by the context-free grammar below (S is the axiom):

$$S \to T\mathtt{a}S_N S_N^\star \mathtt{b}T,\ T \to \mathtt{c},\ T \to TT,\ T \to \varepsilon,$$

$$S_N \to S_{N-1}S_{N-1},\ S_N^\star \to S_{N-1}S_{N-1}^\star, \cdots,\ S_2 \to S_1S_1,\ S_2^\star \to S_1S_1^\star,$$

$$S_1 \to \mathtt{cc},\ S_1^\star \to \mathtt{c}$$

One can check that $\neg \mathrm{L}(p, q, \top)$ whenever there are no children of distance exactly 2^{N+1} such that the first child satisfies p and the second one satisfies q. Note that L is a regular language (even though it is defined with a CFG) and therefore such a property can be also expressed in EML. However, this can be done much more succinctly in EML(CFL) since the context-free grammar is of linear size in N. So, we can conclude that EML(CFL) is at least a succinct version of EML.

Now, let us turn to a genuine context-free property: the sequence of children can be divided into two sequences of equal cardinality such that every child of the first part satisfies p and every child of the second part satisfies q which is a quite natural property to express. In EML(CFL), this statement can be expressed by $\mathrm{L}(p \wedge \neg q, q \wedge \neg p)$ with the context-free language

$$\mathrm{L} = \{\mathtt{a}^i\mathtt{b}^i : i \in \mathbb{N}\}$$

However, it is not difficult to show that the formula $(\sharp p = \sharp q) \wedge \mathrm{L}'(p \wedge \neg q, q \wedge \neg p)$ in EML can also express exactly the same property where $\mathrm{L}' = \mathtt{a}^*\mathtt{b}^*$. In a sense, formulae in EML(CFL) can be viewed as macros for formulae from EML. This is analogous to the situation in modal logic K for which we may consider only one of the operators $\Box$ and $\Diamond$ or both in order to make the formulae clearer. So, EML(CFL) is at least a more friendly version of EML.

The key question remains whether EML(CFL) is strictly more expressive than EML. By generalizing the above reasoning, one can show that context-free constraints of the form $\mathrm{L}(\cdots)$ can be expressed in EML if there are a formula ψ from PrA and a regular language L' such that for every $\alpha \in \Sigma^*$, we have $\alpha \in \mathrm{L}$ iff the Parikh image of α satisfies ψ and $\alpha \in \mathrm{L}'$. Note that $\{\mathtt{a}^i\mathtt{b}^i : i \in \mathbb{N}\}$ precisely satisfies this condition. Such context-free languages L definable by a Presburger formula and by a regular language are exactly the context-free languages definable by so-called *Parikh automata*, see e.g. [8].

The constraints on children that are expressible in EML are Boolean combinations of arithmetical constraints and regular constraints. Such formulae can be reduced equivalently to a disjunction such that each disjunct is a conjunction made of a Boolean combination of arithmetical constraints and made

of a regular constraint of the form $\mathcal{A}(\cdots)$. Indeed, a conjunction of positive or negative regular constraints can be equivalently turned into a single regular constraint. Hence, EML(CFL) is strictly more expressive than EML if there is a context-free language that cannot be recognized by a Parikh automaton. Let us consider the language

$$\mathrm{L}_{REV} = \{\alpha \mathtt{c} \alpha^{\mathrm{rev}} : \alpha \in \{\mathtt{a}, \mathtt{b}\}\}$$

where α^{rev} is the reverse of α. For instance

$$\mathrm{L}_{REV}(p \wedge \neg q \wedge \neg r, q \wedge \neg p \wedge \neg r, r \wedge \neg p \wedge \neg q)$$

roughly states that the sequence of children can be divided into three parts such that the sequence of the first part is the mirror image of the sequence of the third part and the second part is reduce to a single child satisfying $r \wedge \neg p \wedge \neg q$. By adapting the proof of [8, Proposition 3], one can show that L_{REV} is a context-free language that is not definable by a Parikh automaton and therefore EML(CFL) is stricly more expressive that EML, which is after all not a big surprise.

3 On the Expressive Power of EML(VPL)

We show that the logic EML(VPL) has the same expressive power as EML(CFL). It is clear that every VPL is a CFL and therefore EML(CFL) is at least as expressive as EML(VPL). For the other direction, we state that every CFL L over an alphabet Σ can be transformed into a VPL L$'$ over the pushdown alphabet $\langle \Sigma \times \{c\}, \Sigma \times \{i\}, \Sigma \times \{r\}\rangle$ such that $\pi_1(\mathrm{L}') = \mathrm{L}$ where π_1 is the projection over the first component, i.e. π_1 erases the second component whose value is among $\{c, i, r\}$. With this 'embedding' we will be able to show that EML(VPL) is as expressive as EML(CFL) using a truth-preserving reduction.

Theorem 3.1 ([2, Theorem 5.2]) *For any CFL* L *over the alphabet* Σ*, we can effectively build a VPA* M *over the alphabet* $\Sigma_M = \langle \Sigma\times\{c\}, \Sigma\times\{i\}, \Sigma\times\{r\}\rangle$ *such that for all* $\alpha \in \Sigma^*$*, we have* $\alpha \in \mathrm{L}$ *iff there is* $\alpha' \in \mathcal{L}(M)$ *s.t.* $\pi_1(\alpha') = \alpha$*.*

Proof. Let $M' = \langle Q, \Sigma, \Gamma, q_0, \delta, \bot, F\rangle$ be a PDA such that $\mathcal{L}(M') = \mathrm{L}$. Without loss of generality we can also assume that (i) no ε-transitions are used and (ii) the net effect (on the stack) of any transition is null or it pushes or pops one symbol. Define the VPA $M = \langle Q, \Sigma_M, \Gamma, q_0, \delta', \bot, F\rangle$ with $\Sigma_M = \langle \Sigma\times\{c\}, \Sigma\times\{i\}, \Sigma\times\{r\}\rangle$ such that δ' is the minimum relation satisfying

	for every transition $q \to q'$ in M'			there is a transition $q \to q'$ in M		
	reading	*w/stack top*	*action*	*reading*	*w/stack top*	*pushing*
Call	$\mathtt{a}$	$\mathtt{b}'$	push $\mathtt{b}$	$\langle \mathtt{a}, c\rangle$	$\mathtt{b}'$	$\mathtt{b}'\mathtt{b}$
Return	$\mathtt{a}$	$\mathtt{b}'$	pop $\mathtt{b}'$	$\langle \mathtt{a}, r\rangle$	$\mathtt{b}'$	ε
Internal	$\mathtt{a}$	$\mathtt{b}'$	–	$\langle \mathtt{a}, i\rangle$	$\mathtt{b}'$	$\mathtt{b}'$

Intuitively, we simulate the associated PDA by annotating the input word with information on how the stack is managed. The proof follows from the fact that there is a 1-to-1 correspondence between the transitions of M and M'. □

Now we can show that the fragment EML(VPL) is at least as expressive as EML(CFL). This result is surprising because, as we said before, the class of VPLs is strictly included in the class of CFLs. To prove this claim, we give a truth-preserving translation $\mathsf{T} : \mathsf{EML}(\mathrm{CFL}) \to \mathsf{EML}(\mathrm{VPL})$ defined inductively over the terms and formulae. T is defined homomorphically for all Boolean connectives; arithmetical and periodicity constraints are translated as $\mathsf{T}(t > b) \stackrel{\text{def}}{=} \mathsf{T}(t) > b$, $\mathsf{T}(t \equiv_k c) \stackrel{\text{def}}{=} \mathsf{T}(t) \equiv_k c$ and the terms are defined as follows $\mathsf{T}(a \times \sharp\varphi) \stackrel{\text{def}}{=} a \times \sharp\mathsf{T}(\varphi)$, $\mathsf{T}(t + a \times \sharp\varphi) \stackrel{\text{def}}{=} \mathsf{T}(t) + a \times \sharp\mathsf{T}(\varphi)$. The translation is handled differently for regular and context-free languages.

$$\mathsf{T}(\mathrm{L}(\varphi_1, \ldots, \varphi_n)) \stackrel{\text{def}}{=} \mathrm{L}(\mathsf{T}(\varphi_1), \ldots, \mathsf{T}(\varphi_n)) \text{ if } \mathrm{L} \in \mathrm{REG}$$
$$\mathsf{T}(\mathrm{L}(\varphi_1, \ldots, \varphi_n)) \stackrel{\text{def}}{=} \mathrm{L}'(\mathsf{T}(\varphi_1), \mathsf{T}(\varphi_1), \mathsf{T}(\varphi_1), \ldots, \mathsf{T}(\varphi_n), \mathsf{T}(\varphi_n), \mathsf{T}(\varphi_n)) \text{ if } \mathrm{L} \in \mathrm{CFL}.$$

where L' is a VPL built thanks to Theorem 3.1. Note that this is only needed if L is a CFL (hence represented by a PDA) whereas when L is in REG, the translation is homomorphic. The different numbers of arguments are due to the fact that the cardinal of the alphabet of M' is equal to three times the cardinal of Σ. Assuming that the alphabet of M' is grouped by their first component, the above translation assigns the same formula to symbols sharing the same first component. So, $\mathsf{T}(\varphi)$ can be effectively computed from φ. Moreover, an exponential blow-up is observed if formulae are encoded as formula trees, unlike the case when formulae are encoded as directed acyclic graphs.

Lemma 3.2 *Let $\varphi \in \mathsf{EML}(\mathrm{CFL})$. For all models $\mathcal{M} = \langle T, R, (<_s)_{s \in T}, l \rangle$ and nodes $s \in T$, we have $\mathcal{M}, s \models \varphi$ iff $\mathcal{M}, s \models \mathsf{T}(\varphi)$.*

Proof. We prove the result by structural induction. The base case as well as the cases in induction step for Boolean connectives, regularity constraints and arithmetical constraints are by an easy verification. We focus on the remaining case with context-free constraints.

Left to right: Suppose that $\mathcal{M}, s \models \mathrm{L}(\varphi_1, \ldots, \varphi_n)$ and the alphabet of L is $\Sigma = \{\mathtt{a}_1, \ldots, \mathtt{a}_n\}$. There is a word $\alpha = \mathtt{a}_{j_1} \ldots \mathtt{a}_{j_m} \in \mathrm{L}$ and children $s_1, \ldots, s_m$ such that $s_i \models \varphi_{j_i}$ for $i \in [1, m]$. Using the properties satisfied by $\mathrm{L}' \in \mathrm{VPL}$, we know that there is $\alpha' \in \mathrm{L}'$ such that $\pi_1(\alpha') = \alpha$. It is enough to show that for all i, if $\pi_1(\alpha'(i)) = \mathtt{a}_j$ then $s_i \models \mathsf{T}(\varphi_j)$ and this is clearly satisfied by $\mathcal{M}$ by induction hypothesis.

For the other direction, suppose that $\mathcal{M}, s \models \mathsf{T}(\mathrm{L}(\varphi_1, \ldots, \varphi_n))$. Hence, there exist $\alpha' \in \mathrm{L}'$ and children $s_1, \ldots, s_m$. Using again the properties satisfied by $\mathrm{L}' \in \mathrm{VPL}$, it is clear that $\pi_1(\alpha') \in \mathrm{L}$. Observe also that any child s_{j_k} corresponding to a symbol $(\mathtt{a}_{j_k}, \mathtt{action})$ satisfies $s_{j_k} \models \mathsf{T}(\varphi_{j_k})$. With this observation we conclude, by induction hypothesis, that $s_1, \ldots, s_m$ satisfy the needed formulae and thus $\mathcal{M}, s \models \mathrm{L}(\varphi_1, \ldots, \varphi_n)$. □

Observe that the translation T preserves the number of context-free constraints at each depth. Furthermore, polarity of the subformulae is preserved. Consequently, we can state the following result for several fragments.

Theorem 3.3 *There is a reduction from $\mathsf{EML}(\mathrm{CFL})$ [resp. $\mathsf{EML}_i(\mathrm{CFL})$] satisfiability to $\mathsf{EML}(\mathrm{VPL})$ [resp. $\mathsf{EML}_i(\mathrm{VPL})$] satisfiability.*

The exponential blow-up in T for EML(CFL) formulae can be avoided using the renaming technique, whence by introducing new propositional variables. The formula and its translation would not be logically equivalent but they would be equi-satisfiable. Theorem 3.3 states relationships between fragments that will have an important impact on the expressive power and decidability of the logic EML(VPL). Recall that the class of VPLs forms a Boolean algebra and they behave nicely with other logics. Hence, we expected to be able to get numerous positive decidability results as well. From Theorem 3.3, we can conclude that adding VPLs to EML is as powerful as adding the full class of CFLs. In the forthcoming section, we investigate the decidability status of several extensions of EML with CFL-constraints or VPL-constraints.

4 Undecidable Extensions of Presburger Modal Logics

We start by extending EML with CFLs and VPLs and show that they are already undecidable when we allow two positive language constraints to appear simultaneously. Moreover, we show that if we allow negative occurrences of language constraints, we already get undecidability with just one language constraint.

Undecidability of $\mathsf{WEML}_2^+(\mathrm{CFL})$**.** Let $\mathsf{WEML}_2^+(\mathcal{C})$ be the fragment of $\mathsf{EML}_2(\mathcal{C})$ defined as the restriction of the set of formulae specified by

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \mathrm{L}(\psi_1, \ldots, \psi_n)$$

where $\mathrm{L} \in \mathcal{C}$, ψ_i are propositional formulae *and* φ has at most two language constraints from $\mathcal{C}$.Observe that this fragment does not include arithmetical constraints.

Theorem 4.1 *The satisfiability problem for* $\mathsf{WEML}_2^+(\mathrm{CFL})$ *is undecidable.*

Proof. The proof is by reduction from the nonemptiness problem for the intersection of CFLs. Let $\mathrm{L}_1, \mathrm{L}_2$ be two CFLs. We assume that both languages share the same alphabet $\Sigma = \{\mathtt{a}_1, \ldots, \mathtt{a}_n\}$ (if it is not the case, we take Σ as the union of their alphabets). Let φ_i be a formula stating that p_i is the only propositional variable holding among $\{p_1, \ldots, p_n\}$. Let $\psi = \mathrm{L}_1(\varphi_1, \ldots, \varphi_n) \wedge \mathrm{L}_2(\varphi_1, \ldots, \varphi_n)$. We prove that ψ is satisfiable iff $\mathrm{L}_1 \cap \mathrm{L}_2 \neq \emptyset$.

For the 'if' direction suppose that $\mathcal{M}, s \models \psi$ and there exist a sequence of children $s_1, \ldots, s_k$ and words $\alpha_1 \in \mathrm{L}_1$, $\alpha_2 \in \mathrm{L}_2$ such that $|\alpha_1| = |\alpha_2| = k$. Let $w_1 = \mathtt{a}_{i_1} \ldots \mathtt{a}_{i_k}$ and $w_2 = \mathtt{a}_{j_1} \ldots \mathtt{a}_{j_k}$. Note that for $\ell \in \{1, 2\}$, $\alpha_\ell(i) = \mathtt{a}_j$ iff $s_i \models p_j$. As only one propositional variable can be true of a given child then we can conclude that $\alpha_1 = \alpha_2$ and therefore $\mathrm{L}_1 \cap \mathrm{L}_2 \neq \emptyset$. For the 'only if' direction: Let $\alpha \in \mathrm{L}_1 \cap \mathrm{L}_2$ and $|\alpha| = k$. Take the model $\mathcal{M}, s$ with children $s_1, \ldots, s_k$ such that $s_i \models p_j$ iff $\alpha(i) = \mathtt{a}_j$. It is easy to check that this model satisfies ψ. □

Undecidability of $\mathsf{EML}_2(\mathrm{VPL})$**.** We show that the satisfiability problem for EML(VPL) is undecidable. Moreover, the EML_2(VPL) fragment is already undecidable. In contrast with CFLs, there is one more parameter to be taken

into account in our analysis: every VPL is defined over a pushdown alphabet. Therefore, given many VPLs occurring in a formula, we analyze how the relationship among their partitions relates to the undecidability result.

Corollary 4.2 *The satisfiability problem for* $\mathsf{WEML}_2^+(\mathrm{VPL})$ *is undecidable, even restricted to formulae such that every two distinct VPLs occurring at the same depth share the same pushdown alphabet.*

Proof. Undecidability is by reduction from the satisfiability problem for the fragment $\mathsf{WEML}_2^+(\mathrm{CFL})$ (see Theorem 4.1). Given a formula $\varphi \in \mathsf{WEML}_2^+(\mathrm{CFL})$, we can use the translation T to obtain a formula $\mathsf{T}(\varphi)$ in $\mathsf{WEML}_2^+(\mathrm{VPL})$. By truth-preservation of T, we obtain that the satisfiability problem for $\mathsf{WEML}_2^+(\mathrm{VPL})$ is undecidable. Observe that we can always assume that for $\varphi \in \mathsf{WEML}_2^+(\mathrm{CFL})$, if two distinct context-free languages occur at the same modal depth, then they share the same pushdown alphabet. In this way, we get a translation $\mathsf{T}(\varphi)$ in the fragment mentioned in the statement of the theorem. □

Considering T, the translation of a CFL provides the same pushdown alphabet and formula arguments. One could argue that it is exactly this connection through the pushdown alphabet that allows us to encode the nonemptiness problem for the intersection of CFLs. In the following theorem we show that, even if the VPLs do not share the pushdown alphabet, we can still encode the same problem using propositional variables that play the role of 'binders'.

Theorem 4.3 *The satisfiability problem for* $\mathsf{WEML}_2^+(\mathrm{VPL})$ *is undecidable even restricted to formulae such that every two distinct VPLs occuring at the same modal depth do not share the pushdown alphabet.*

Proof. Let $\mathrm{L}_1, \mathrm{L}_2$ be CFLs over the alphabet Σ. By Theorem 3.1, one can build two VPA M_1 and M_2 over the pushdown alphabet $\langle \Sigma \times \{c\}, \Sigma \times \{i\}, \Sigma \times \{r\} \rangle$ such that for all words $\alpha \in \Sigma^*$, we have (1) $\alpha \in \mathrm{L}_1$ iff there is $\alpha' \in \mathcal{L}(M_1)$ such that $\pi_1(\alpha') = \alpha$ and (2) $\alpha \in \mathrm{L}_2$ iff there is $\alpha' \in \mathcal{L}(M_2)$ such that $\pi_1(\alpha') = \alpha$.

Let Σ_1 and Σ_2 be two distinct alphabets of cardinality $3 \times \mathrm{card}(\Sigma)$ with $\Sigma_1 \cap \Sigma_2 = \emptyset$ and, $\sigma_1 : \Sigma \times \{c, r, i\} \to \Sigma_1$ and $\sigma_2 : \Sigma \times \{c, r, i\} \to \Sigma_2$ be two bijective renamings. Recall that VPLs are closed under renamings and one can easily compute VPA $M_1^\star$ and $M_2^\star$ such that $\mathcal{L}(M_1^\star) = \sigma_1(\mathcal{L}(M_1))$ and $\mathcal{L}(M_2^\star) = \sigma_2(\mathcal{L}(M_2))$. Note that $M_1^\star$ and $M_2^\star$ are defined over distinct pushdown alphabets. Again, for all words $\alpha \in \Sigma^*$, we have (1′) $\alpha \in \mathrm{L}_1$ iff there is $\alpha' \in \mathcal{L}(M_1^\star)$ such that $\pi_1(\sigma_1^{-1}(\alpha')) = \alpha$ and (2′) $\alpha \in \mathrm{L}_2$ iff there is $\alpha' \in \mathcal{L}(M_2^\star)$ such that $\pi_1(\sigma_2^{-1}(\alpha')) = \alpha$.

Let $\Sigma = \{\mathtt{a}_1, \ldots, \mathtt{a}_n\}$, $\Sigma_1 = \{\mathtt{s}_1, \ldots, \mathtt{s}_{3n}\}$ and $\Sigma_2 = \{\mathtt{s}'_1, \ldots, \mathtt{s}'_{3n}\}$. Let φ_i be a formula that states that the only propositional variable true among $\{p_1, \ldots, p_{3n}\}$ at a given node is p_i. We define a function which identifies each symbol from the original alphabet Σ with a propositional variable. Let $g : \Sigma \times \{c, r, i\} \to \{\varphi_1, \ldots, \varphi_{3n}\}$ be defined as $g((\mathtt{a}_i, \mathtt{action})) = \varphi_i$ and let $\theta_j^i = g(\sigma_i^{-1}(\mathtt{s}_j))$ then we define $\psi = \mathcal{L}(M_1^\star)(\theta_1^1, \ldots, \theta_{3n}^1) \wedge \mathcal{L}(M_2^\star)(\theta_1^2, \ldots, \theta_{3n}^2)$. It is easy to prove that ψ is satisfiable iff $\mathcal{L}(M_1) \cap \mathcal{L}(M_2) \neq \emptyset$ using an argu-

ment similar to that of Theorem 4.1. We conclude that $\mathrm{L}_1 \cap \mathrm{L}_2 \neq \emptyset$ iff ψ is satisfiable. □

Undecidability of $\mathsf{WEML}_1^-(\mathrm{CFL})$**.** Let $\mathsf{WEML}_1^-(\mathrm{CFL})$ be the restriction of $\mathsf{EML}_1(\mathrm{CFL})$ defined as a fragment specified by

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \neg \mathrm{L}(p_1, \ldots, p_n) \mid \Box \psi$$

where $\mathrm{L} \in \mathrm{CFL}$, ψ is a propositional formula *and* φ has at most one context-free constraint. Given a CFL L over the alphabet $\Sigma = \{\mathtt{a}_1, \ldots, \mathtt{a}_k\}$, we build a formula φ_{L} in $\mathsf{WEML}_1^-(\mathrm{CFL})$ such that $\mathrm{L} \neq \Sigma^*$ iff φ_{L} is satisfiable. Since the universality problem for CFLs is undecidable (see e.g. [19]) we conclude that the satisfiability problem for $\mathsf{WEML}_1^-(\mathrm{CFL})$ is undecidable too. Let $\varphi_{\mathrm{uni}}(p_1, \ldots, p_k)$ be a formula in $\mathsf{WEML}_1^-(\mathrm{CFL})$ enforcing that any child satisfies exactly one propositional variable among $\{p_1, \ldots, p_k\}$, we define

$$\varphi_{\mathrm{L}} = \varphi_{\mathrm{uni}}(p_1, \ldots, p_k) \wedge \neg \mathrm{L}(p_1, \ldots, p_k)$$

Theorem 4.4 *The satisfiability problem for* $\mathsf{WEML}_1^-(\mathrm{CFL})$ *is undecidable.*

Proof. First, suppose that $\mathrm{L} \neq \Sigma^*$; hence there is a word $w = \mathtt{a}_{i_1} \cdots \mathtt{a}_{i_n} \notin \mathrm{L}$. Let us consider the model $\mathcal{M}$ with a root node s and children $s_1 < \cdots < s_n$ such that for every $j \in [1, n]$, $s_j \models \varphi_{i_j}$. It is clear that $\mathcal{M}, s \models \varphi_{\mathrm{uni}}(p_1, \ldots, p_k)$. The crucial observation is that, given a model where exactly one propositional variable holds at each child, forming the sequence $p_{i_1}, \ldots, p_{i_n}$ then $\mathcal{M}, s \models \mathrm{L}(p_1, \ldots, p_k)$ iff $\mathtt{a}_{i_1} \cdots \mathtt{a}_{i_n} \in \mathrm{L}$. Hence, $\mathcal{M}, s \models \neg \mathrm{L}(p_1, \ldots, p_k)$.

Now suppose that there is a model $\mathcal{M}$ and a node s such that $\mathcal{M}, s \models \varphi_{\mathrm{L}}$. If s has no successor, then $\epsilon \notin \mathrm{L}$, otherwise $\mathcal{M}, s \models \mathrm{L}(p_1, \ldots, p_k)$. In that case $\mathrm{L} \neq \Sigma^*$. Now suppose that s has children $s_1 < \ldots < s_n$ with $n \geqslant 1$. As before, the propositional variables true at the children form a sequence $p_{i_1}, \ldots, p_{i_n}$ and $\mathcal{M}, s \models \mathrm{L}(p_1, \ldots, p_k)$ iff $\mathtt{a}_{i_1} \cdots \mathtt{a}_{i_n} \in \mathrm{L}$. Therefore there exists a word $\mathtt{a}_{i_1} \cdots \mathtt{a}_{i_n} \notin \mathrm{L}$ which implies $\mathrm{L} \neq \Sigma^*$. □

As a corollary, the satisfiability problem for $\mathsf{EML}_1^-(\mathrm{VPL})$ is undecidable too.

5 Decidability of $\mathsf{EML}_1^+(\mathrm{CFL})$

We saw that it is enough to have two positive occurrences (i.e., with nonzero polarity) of CFL-constraints or one negative occurrence (i.e., with zero polarity) of a CFL-constraint to have an undecidable satisfiability problem. It remains one interesting extension of EML to be studied: let $\mathsf{EML}_1^+(\mathrm{CFL})$ be the fragment in which there is at most one CFL-constraint at each depth, occurring positively. $\mathsf{EML}_i^+(\mathcal{C})$ is defined as the subset of $\mathsf{EML}_i(\mathcal{C})$ such that every formula's tree node labelled by a formula of the form

1. $\mathrm{L}(\cdots)$ with $\mathrm{L} \in \mathcal{C}$ has a positive polarity,
2. $\sum_i a_i \sharp \varphi_i > b$ with a positive polarity has all $a_i, b > 0$ or all $\varphi_i \in \mathsf{EML}(\mathrm{REG})$,
3. $\sum_i a_i \sharp \varphi_i > b$ with a negative polarity has each $\varphi_i \in \mathsf{EML}(\mathrm{REG})$,

4. $\sum_i a_i \sharp \varphi_i \equiv_k c$ has $\varphi_i \in \mathsf{EML}(\mathrm{REG})$.

For example, $\mathrm{L}_3(p,q) \wedge \mathrm{L}_1(\mathrm{L}_3(p,q),q) \in \mathsf{EML}_1^+(\mathrm{CFL})$ with $\mathrm{L}_1 \in \mathrm{REG}$, $\mathrm{L}_3 \in \mathrm{CFL}$. We show that the satisfiability problem for $\mathsf{EML}_1^+(\mathrm{CFL})$ is decidable, which is the main decidability result of the paper. Even though the above definition has been finely tailored for our proof to work, we think that it is probably the only potentially interesting fragment left to be analyzed. Nonetheless, this fragment adds extra power over EML: suppose you have an XML file containing 'buy' and 'sell' records. With $\mathsf{EML}_1^+(\mathrm{CFL})$ you can require that for every 'sell' there is a matching 'buy' record before it whereas that cannot be done in EML.

We provide a decision procedure for $\mathsf{EML}_1^+(\mathrm{CFL})$ generalizing the one for EML [13]. A major difference is that we have to deal with CFL-constraints with a fine-tuned procedure. Whereas EML satisfiability is in PSPACE, our algorithm for $\mathsf{EML}_1^+(\mathrm{CFL})$ requires strictly more space since we need to store the stack content which can be of double exponential length in the worst-case, preventing us from using an on-the-fly algorithm as done for EML. Our decidability result is established thanks to a Ladner-like algorithm [22] (see also [29,13]). The original algorithm from [22] allows to show that the satisfiability problem for modal logic K is in PSPACE and it can be viewed as a tableaux-like procedure with non-determinism but without tableaux rules. The beauty of Ladner-like algorithms rests on their simple structure with fine-tuned recursivity; moreover it does not use any specific formalism such as tableaux, automata, sequents etc. Our algorithm below extends the one presented in [13] for the satisfiability of the basic EML. We start by defining the closure for finite sets of formulae. Intuitively, the closure $\mathrm{cl}(X)$ of X contains all formulae useful to evaluate the truth of formulae in X.

Definition 5.1 Let X be a finite set of formulae, $\mathrm{cl}(X)$ is the smallest set satisfying

- $X \subseteq \mathrm{cl}(X)$, $\mathrm{cl}(X)$ is closed under subformulae,
- if $\psi \in \mathrm{cl}(X)$, then $\neg\psi \in \mathrm{cl}(X)$ (we identify $\neg\neg\psi$ with ψ),
- let $\mathtt{K}$ be the least common multiple (lcm) of all the constants k occurring in subformulae of the form $t \equiv_k c$. If $t \equiv_k c \in \mathrm{cl}(X)$, then $t \equiv_{\mathtt{K}} c' \in \mathrm{cl}(X)$ for every $c' \in [0, \mathtt{K}-1]$.

Observe that, since $\mathsf{EML}_1^+(\mathrm{CFL})$ is not closed under negation, formulae in $\mathrm{cl}(X)$ may not belong to $\mathsf{EML}_1^+(\mathrm{CFL})$ even if X is a set of $\mathsf{EML}_1^+(\mathrm{CFL})$ formulae. A set X of formulae is said to be closed iff $\mathrm{cl}(X) = X$. We refine the notion of closure by introducing a new parameter n: each set $\mathrm{cl}(n,\varphi)$ is a subset of $\mathrm{cl}(\{\varphi\})$ that corresponds to a closed set obtained from subformulae of depth n. We have $\mathrm{cl}(0,\varphi) \supseteq \mathrm{cl}(1,\varphi) \supseteq \mathrm{cl}(2,\varphi) \cdots$ and the sets are defined by peeling of modalities layer by layer.

Definition 5.2 Let $\varphi \in \mathsf{EML}_1^+(\mathrm{CFL})$; for $n \in \mathbb{N}$, $\mathrm{cl}(n,\varphi)$ is the smallest set such that

- $\mathrm{cl}(0,\varphi) = \mathrm{cl}(\{\varphi\})$, for every $n \in \mathbb{N}$, $\mathrm{cl}(n,\varphi)$ is closed,

- for all $n \in \mathbb{N}$ and $\sharp\psi$ occurring in some formula of $\mathrm{cl}(n,\varphi)$, $\psi \in \mathrm{cl}(n+1,\varphi)$,
- for all $n \in \mathbb{N}$ and $\mathrm{L}(\varphi_1,\ldots,\varphi_m) \in \mathrm{cl}(n,\varphi)$, then $\{\varphi_1,\ldots,\varphi_m\} \subseteq \mathrm{cl}(n+1,\varphi)$.

We will concentrate on subsets of $\mathrm{cl}(n,\varphi)$ whose conjunction of elements is $\mathsf{EML}_1^+(\mathrm{CFL})$ satisfiable. A necessary condition to be satisfiable is to be consistent locally, i.e. at the propositional level and at the level of arithmetical constraints. Let $\mathtt{K}$ be the lcm of all the constants k occurring in subformulae of φ of the form $t \equiv_k c$.

Definition 5.3 A set $X \subseteq \mathrm{cl}(n,\varphi)$ is said to be n-locally consistent iff:

- if $\neg\psi \in \mathrm{cl}(n,\varphi)$, then $\neg\psi \in X$ iff $\psi \notin X$,
- if $\psi_1 \wedge \psi_2 \in \mathrm{cl}(n,\varphi)$, then $\psi_1 \wedge \psi_2 \in X$ iff $\psi_1, \psi_2 \in X$,
- if $\psi_1 \vee \psi_2 \in \mathrm{cl}(n,\varphi)$, then $\psi_1 \vee \psi_2 \in X$ iff $\psi_1 \in X$ or $\psi_2 \in X$,
- if $t \equiv_k c \in \mathrm{cl}(n,X)$, then there is a unique $c' \in [0,\mathtt{K}-1]$ such that $t \equiv_{\mathtt{K}} c' \in X$,
- if $t \equiv_k c \in \mathrm{cl}(n,X)$, then $\neg t \equiv_k c \in X$ iff there is $c' \in [0,\mathtt{K}-1]$ such that $t \equiv_{\mathtt{K}} c' \in X$ and not $c' \equiv_k c$,

The *kernel* of an n-locally consistent set X is $\ker(X) = X \cap \mathsf{EML}_1^+(\mathrm{CFL})$. As we have observed, the closure operation may introduce formulae outside $\mathsf{EML}_1^+(\mathrm{CFL})$. Negation of EML formulae is not a problem (since they are also in EML) whereas negation of $\mathsf{EML}_1^+(\mathrm{CFL})$ formulae may lead to undecidability (e.g., because of a negative polarity). That is why, in our algorithm, we shall only try to satisfy formulae from $\ker(X)$ and this shall be sufficient from the way the fragment $\mathsf{EML}_1^+(\mathrm{CFL})$ is designed. A slight variation in the definition of $\mathsf{EML}_1^+(\mathrm{CFL})$ may lead to undecidability of the satisfiability problem. Regarding size, observe that $\mathrm{card}(\mathrm{cl}(X))$ is exponential in $\mathrm{card}(X)$. Nevertheless, consistent sets of formulae that are satisfiable contain exactly one formula from the set $\{t \equiv_{\mathtt{K}} c : c \in [0,\mathtt{K}-1]\}$ for each constraint $t \equiv_k c'$ in X. Hence, as explained in [13], encoding consistent sets will only require polynomial space.

Definition 5.4 Let φ be an $\mathsf{EML}_1^+(\mathrm{CFL})$ formula, N be a natural number and $\mathcal{M}$ be a finite tree model such that $\mathcal{M}, s \models \varphi$ for some node s. We say that $\langle \mathcal{M}, s\rangle$ is *N-bounded* for φ iff for every node s' of distance d from s, the number of successors of s' is bounded by $nb(d+1) \times N$ where $nb(d+1)$ is the number of distinct $(d+1)$-locally consistent sets (with respect to φ).

We define the function SAT (see below) such that φ is $\mathsf{EML}_1^+(\mathrm{CFL})$ satisfiable in some N-bounded model iff there is $X \subseteq \mathrm{cl}(0,\varphi)$ such that $\varphi \in \ker(X)$ and $\mathrm{SAT}(X,0)$ has a computation returning `true`. Indeed, the function $\mathrm{SAT}(X,d)$ is globally parameterized by some natural number N and by the formula φ. These two parameters should be understood as global variables. The main difference with the algorithm in [13] is related to the update of the stack and we restrict ourselves to formulae in $\ker(X)$.

`function` SAT(X, d)

(consistency) if X is not d-locally consistent then `abort`;

(base case) if X contains only propositional formulae then return `true`;

(initialization-counters) for every $\psi \in \ker(\mathrm{cl}(d+1, \varphi))$ that is not a periodicity constraint of the form $t \equiv_{\mathtt{K}} c$, $C_\psi := 0$;

(initialization-fsa) for every FSA $\mathcal{A}(\psi_1, \ldots, \psi_\alpha) \in \ker(X)$, the state variable $q_{\mathcal{A}(\psi_1,\ldots,\psi_\alpha)} := q_0$ for some initial state q_0 of $\mathcal{A}$;

(initialization-fsa-complement) for every FSA $\neg\mathcal{A}(\psi_1, \ldots, \psi_\alpha) \in \ker(X)$, $Z_{\neg\mathcal{A}(\psi_1,\ldots,\psi_\alpha)} := I$ where I is the set of initial states of $\mathcal{A}$;

(initialization-pda) for every CFL-constraint $\mathcal{C}(\psi_1, \ldots, \psi_\alpha) \in \ker(X)$ where $\mathcal{C}$ is a PDA, the state variable $q_{\mathcal{C}(\psi_1,\ldots,\psi_\alpha)} := q_0$ for some initial state q_0 of $\mathcal{C}$ and the stack variable $S_{\mathcal{C}(\psi_1,\ldots,\psi_\alpha)} := \bot$ for the initial stack symbol $\bot$ of $\mathcal{C}$;

(guess-number-children) guess NB in $\{0, \ldots, nb(d+1) \times N\}$;

(guess-children-from-left-to-right) for $i = 1$ to NB do

(i) guess $x \in \{1, \ldots, nb(d+1)\}$;

(ii) if not SAT$(Y_x, d+1)$ then `abort`;

(iii) for every $\psi \in \ker(\mathrm{cl}(d+1, \varphi))$ different from some $t \equiv_K c$ such that $\psi \in Y_x$ do $C_\psi := C_\psi + 1$;

(iv) for every finite state automaton $\mathcal{A}(\psi_1, \ldots, \psi_\alpha) \in \ker(X)$,

a. guess a transition $q_{\mathcal{A}(\psi_1,\ldots,\psi_\alpha)} \xrightarrow{a_i} q'$ in $\mathcal{A}$ with $\Sigma_{\mathcal{A}} = \mathtt{a}_1, \ldots, \mathtt{a}_\alpha$;

b. if $\psi_i \in Y_x$, then $q_{\mathcal{A}(\psi_1,\ldots,\psi_\alpha)} := q'$, otherwise `abort`;

(v) for every finite state automaton $\neg\mathcal{A}(\psi_1, \ldots, \psi_\alpha) \in \ker(X)$, $Z_{\neg\mathcal{A}(\psi_1,\ldots,\psi_\alpha)} := \{q : \exists\, q' \in Z_{\neg\mathcal{A}(\psi_1,\ldots,\psi_\alpha)}, q' \xrightarrow{\mathtt{a}_i} q, \psi_i \in Y_x\}$;

(vi) for every CFL-constraint $\mathcal{C}(\psi_1, \ldots, \psi_\alpha) \in \ker(X)$,

a. let $S_{\mathcal{C}(\psi_1,\ldots,\psi_\alpha)} = \mathtt{b}_1 \ldots \mathtt{b}_k$ be the stack content of $\mathcal{C}$, guess a transition $q_{\mathcal{C}(\psi_1,\ldots,\psi_\alpha)} \xrightarrow{\mathtt{a}_i} q'$ poping $\mathtt{b}_1$ and pushing γ in $\mathcal{C}$ with $\Sigma_{\mathcal{C}} = \mathtt{a}_1, \ldots, \mathtt{a}_\alpha$;

b. if $\psi_i \in Y_x$, then $q_{\mathcal{A}(\psi_1,\ldots,\psi_\alpha)} := q'$ and $S_{\mathcal{C}(\psi_1,\ldots,\psi_\alpha)} := \gamma\mathtt{b}_2 \ldots \mathtt{b}_k$, otherwise `abort`;

(final-checking)

(i) for every $\sum_i a_i \sharp \psi_i \sim b \in X$, if $\sum_i a_i \times C_{\psi_i} \sim b$ does not hold, then `abort`,

(ii) for every $\sum_i a_i \sharp \psi_i \equiv_k c \in X$, if $\sum_i a_i \times C_{\psi_i} \equiv_k c$ does not hold, `abort`,

(iii) for every (either finite-state or pushdown) automaton $\mathcal{A}(\psi_1, \ldots, \psi_\alpha) \in X$, if $q_{\mathcal{A}(\psi_1,\ldots,\psi_\alpha)}$ is not a final state of $\mathcal{A}$, then `abort`;

(iv) for every FSA $\neg\mathcal{A}(\psi_1, \ldots, \psi_\alpha) \in X$, if $Z_{\neg\mathcal{A}(\psi_1,\ldots,\psi_\alpha)}$ contains a final state of $\mathcal{A}$, then `abort`;

(return-true) return `true`.

We shall later fix N which will be doubly exponential in $|\varphi|$ (see Lemma 5.9). The first argument X is intended to be a subset of $\mathrm{cl}(d, \varphi)$ and the $(d+1)$-locally consistent sets are denoted by Y_i for some $1 \leqslant i \leqslant nb(d+1)$. A call to SAT$(X, d)$ performs the following actions. First it checks whether X is d-locally consistent and if the modal degree is zero, then it returns `true` in case of d-locally consistency. In order to check that $\ker(X)$ is satisfiable, children of the node are guessed from left to right (providing an ordering of the successors). For each $\psi \in \ker(X)$, there is a counter C_ψ which contains the current

number of children that should satisfy ψ. Similarly, we keep track of periodicity constraints. For each subformula in $\ker(X)$ whose outermost connective is automata-based, we introduce a variable that encodes the current state and stack content. At the end of the guess of the children, this variable should be equal to a final state of the automaton. By contrast, for each formula in $\ker(X)$ whose outermost connective is the negation of some automata-based formula, we introduce a variable that encodes the set of states that could be reached so far in the automaton (simulating a subset construction of the underlying automaton). We never have a negation of a CFL-constraint. At the end of the guess of the children, this variable should not contain any final state of the automaton. After guessing at most $N \times nb(d+1)$ children, there is a final checking which verifies that periodicity and arithmetical constraints are satisfied. Keeping track of the stack may hurt a *lot*, i.e., the resulting size could be linear on the number of children. Therefore, of exponential space.

Lemma 5.5 *For all 0-locally consistent sets X and computations of* $\mathrm{SAT}(X, 0)$

(i) *the recursive depth is linear in $|\varphi|$,*

(ii) *each call requires space polynomial in the sum of the space for encoding 0-locally consistent sets and $N + 2^{|\varphi|}$,*

Proof. Since $\mathrm{cl}(|\varphi|, \varphi) = \emptyset$, the size of the stack of recursive calls to SAT is at most $|\varphi|$. In the function SAT, the steps (consistency), (base case), (initialization-counters), (initialization-fsa), (initialization-fsa-complement) and (initialization-pda) can be obviously checked in polynomial time in φ (and therefore in polynomial space). Indeed, every n-locally consistent set has cardinal at most $2 \times |\varphi|$ and can be encoded with a polynomial amount of bits with respect of $|\varphi|$; moreover given $X \subseteq \mathrm{cl}(0, \varphi)$ of cardinal at most $2 \times |\varphi|$ and $n \in \mathbb{N}$, one can decide in polynomial-time in $|\varphi|$ whether X is n-locally consistent. In the step (guess-children-from-left-to-right), one needs a counter to count at most until $nb(d+1) \times N$. A polynomial amount of bits in $|\varphi| + log(N)$ suffices. All the non-recursive instructions in (guess-children-from-left-right) can be done in time polynomial in $|\varphi| + log(N)$. Since at the end of the step (guess-children-from-left-right), the values of the counters are less than or equal to $nb(d+1) \times N$, checking the points (i) and (ii) in (final-checking) can be done in polynomial space in $|\varphi| + log(N)$ (remember that the encoding of constants a_i, b and c and k are already in linear space in $|\varphi|$). The only variables that requires polynomial space in $2^{|\varphi|} + N$ are those containing stack contents since the number of children NB is bounded by $nb(d+1) \times N$.□

An analogous of [13, Lemma 5] is shown below.

Lemma 5.6 *Let $X \subseteq \mathrm{cl}(0, \varphi)$, if* $\mathrm{SAT}(X, 0)$ *has successful computation and $\varphi \in \ker(X)$, then φ is* $\mathsf{EML}_1^+(\mathrm{CFL})$ *satisfiable in an N-bounded model.*

Proof. Assume that $\mathrm{SAT}(X, 0)$ has an accepting computation with $\varphi \in \ker(X)$. Let us build a model $\mathcal{M} = \langle T, R, (<_s)_{s \in T}, l \rangle$ for which there is $s \in T$ such that for every $\psi \in \ker(X)$ we have $\mathcal{M}, s \models \psi$.

From an accepting computation of $\mathrm{SAT}(X, 0)$, we consider the following finite ordered tree $\langle T, R, (<_s)_{s\in T}, \ell\rangle$ that corresponds to the calls tree of $\mathrm{SAT}(X, 0)$.

- $\langle T, R, (<_s)_{s\in T}\rangle$ is a finite ordered tree,
- for each $s \in T$, $\ell(s) = \langle Y, d\rangle$ for some d-consistent set Y,
- the root node s_0 is labelled by $\langle X, 0\rangle$,
- for each node s with $s_1 <_s \cdots <_s s_n$, the call related to $\ell(s)$ recursively calls SAT with the respective arguments $\ell(s_1)$, ..., $\ell(s_n)$ and in this very ordering.

The model $\mathcal{M}$ we are looking for, is precisely $\mathcal{M} = \langle T, R, (<_s)_{s\in T}, l\rangle$ for which $l(s) = Y \cap \mathrm{AT}$ where $\ell(s) = \langle Y, d\rangle$ for each s.

By structural induction on ψ, we shall show that for all $s \in T$ with labeling $\ell(s) = \langle Y, d\rangle$ we have, $\psi \in \ker(Y)$ implies $\mathcal{M}, s \models \psi$. Consequently, we then get $\mathcal{M}, s_0 \models \varphi$. The case when ψ is a propositional variable is by definition of l.

Induction hypothesis: for all ψ such that $|\psi| \leqslant n$, and for all $s \in T$ with $\ell(s) = \langle Y, d\rangle$, if $\psi \in \ker(Y)$ then $\mathcal{M}, s \models \psi$.

Let ψ be a formula such that $|\psi| = n + 1$.

Basic boolean cases: $\psi = \psi_1 \wedge \psi_2$ and $\psi = \psi_1 \vee \psi_2$.
Observe that if $\psi \in \ker(Y)$ then $\psi_1, \psi_2 \in \ker(Y)$. These cases follow easily with this observation and the definition of n-locally consistent set.

Negation of boolean operators: $\psi = \neg(\psi_1 \wedge \psi_2)$ and $\psi = \neg(\psi_1 \vee \psi_2)$
For the first case, by definition of n-locally consistent set, at least one of $\neg\psi_1$ or $\neg\psi_2$ belongs to Y. Suppose that $\neg\psi_1 \in Y$. It is easy to check that, as $\psi \in \ker(Y)$ we also have $\neg\psi_1 \in \ker(Y)$. By the induction hypothesis, we get that $\mathcal{M}, s \models \neg\psi_1$ and therefore $\mathcal{M}, s \models \psi$. The case for $\psi = \neg(\psi_1 \vee \psi_2)$ is similar.

Positive language: $\psi = \mathrm{L}(\psi_1, \ldots, \psi_k)$ with $\mathrm{L} \in \mathrm{REG} \cup \mathrm{CFL}$.
Let $s \in T$ with $\ell(s) = \langle Y, d\rangle$ such that $\psi \in \mathrm{cl}(d, \varphi)$. By definition of T, $\mathrm{SAT}(Y, d)$ has an accepting computation. If $\psi \in \ker(Y)$, then each call in the sequence $\mathrm{SAT}(Y_{x_1}, d+1)$, ..., $\mathrm{SAT}(Y_{x_{\mathrm{NB}}}, d+1)$ has an accepting computation. Hence the children of s are, from left to right: $s_1, \ldots, s_{\mathrm{NB}}$ such that $\ell(s_i) = \langle Y_{x_i}, d + 1\rangle$. Then, it is not difficult to show that the steps (initialization-fsa), (initialization-pda), (guess-children-from-left-to-right) and (final-checking)(iii) guarantee that $\mathcal{M}, s \models \psi$ by using the induction hypothesis and the fact that each ψ_i belongs to $\ker(Y)$ too.

Negation of regular language: $\psi = \neg\mathrm{L}(\psi_1, \ldots, \psi_k)$ with $\mathrm{L} \in \mathrm{REG}$.
Let $s \in T$ with $\ell(s) = \langle Y, d\rangle$ such that $\psi \in \mathrm{cl}(d, \varphi)$. By definition of T, $\mathrm{SAT}(Y, d)$ has an accepting computation. Suppose s has no children, then $\varepsilon \notin \mathrm{L}$, otherwise (final-checking) would fail. This implies $\mathcal{M}, s \not\models \mathrm{L}(\psi_1, \ldots, \psi_k)$.

Suppose s has children $s_1, \ldots, s_{\mathrm{NB}}$ such that $\ell(s_i) = \langle Y_{x_i}, d + 1\rangle$. Given an arbitrary word $\gamma = \psi_{j_1} \ldots \psi_{j_{\mathrm{NB}}} \in \mathrm{L}$ suppose that $\psi_{j_i} \in Y_{j_i}$ for all i. In this case the step (guess-children-from-left-to-right) would finish with a set of

states including a final state and therefore (final-checking) would abort. Hence, there exists $\psi_{j_k} \notin Y_{j_k}$. By the definition of n-locally consistent set, this implies that $\neg\psi_{j_k} \in Y_{j_k}$. As $\psi \in \mathsf{EML}_1^+(\mathrm{CFL})$ and going through language constraints preserve polarity we know that $\neg\psi_{j_k} \in \mathsf{EML}_1^+(\mathrm{CFL})$ too. We conclude that $\neg\psi_{j_k} \in \ker(Y_{j_k})$ and using the induction hypothesis it is easy to see that $\mathcal{M}, s_k \not\models \psi_{j_k}$ and therefore $\mathcal{M}, s \not\models \mathrm{L}(\psi_1, \ldots, \psi_k)$.

Arithmetical constraint: $\psi = \sum_{i=1}^{i=\alpha} a_i \sharp \psi_i > b$.
Let $s \in T$ such that $\ell(s) = \langle Y, d\rangle$ and $\psi \in \mathrm{cl}(d, \varphi)$. By definition of T, $\mathrm{SAT}(Y, d)$ has an accepting computation. If $\psi \in \ker(Y)$, then each call in the sequence $\mathrm{SAT}(Y_{x_1}, d+1)$, $\ldots$, $\mathrm{SAT}(Y_{x_{NB}}, d+1)$ has an accepting computation. For every $i \in [1, \alpha]$, there are exactly C_{ψ_i} elements in $Y_{x_1}, \ldots, Y_{x_{NB}}$ that contain ψ_i where C_{ψ_i} is the value of the counter after the step (guess-children-from-left-to-right) in the above-mentioned successful computation for $\mathrm{SAT}(Y, d)$. Hence the children of s in $\mathcal{M}$ are the following (from left to right): $s_1, \ldots, s_{NB}$ with $\ell(s_i) = \langle Y_{x_i}, d+1\rangle$.

Case 1: All the subformulae ψ_i belong to EML.
There are exactly C_{ψ_i} children satisfying ψ_i by induction hypothesis and by the fact that either ψ_i or $\neg\psi_i$ belongs to Y_{x_i}. The sum of the terms will be at least $b+1$ and together with the steps (initialization-counters), (guess-children-from-left-to-right) and (final-checking) guarantee that $\mathcal{M}, s \models \psi$.

Case 2: Some formula ψ_i does not belong to EML.
There are at least C_{ψ_i} children satisfying ψ_i by induction hypothesis. Observe that, given a formula ψ_i belonging to some set $\ker(Y_{x_j})$ we know that $\mathcal{M}, s_j \models \psi_i$. On the contrary, if $\psi_i \notin \ker(Y_{x_j})$ we cannot say that $\mathcal{M}, s_j \not\models \psi_i$. Therefore, the values C_{ψ_i} are *lower bounds* for the number of children satisfying ψ_i. Since each a_i is strictly greater than zero, the sum of the terms will be at least $b+1$ and together with the steps (initialization-counters), (guess-children-from-left-to-right) and (final-checking) guarantee that $\mathcal{M}, s \models \psi$.

Negation of arithmetical constraint: $\psi = \neg \sum_{i=1}^{i=\alpha} a_i \sharp \psi_i > b$.
First, observe that if $\psi \in \mathsf{EML}_1^+(\mathrm{CFL})$, then all $\psi_i \in \mathsf{EML}(\mathrm{REG})$. In this case we need to show that $\mathcal{M}, s \models \sum_{i=1}^{i=\alpha} a_i \sharp \psi_i \leqslant b$, which can be done with an argument similar to the case of positive constraints.

Periodicity constraints: $\psi = \sum_{i=1}^{i=\alpha} a_i \sharp \psi_i \equiv_k c$ and $\psi = \neg \sum_{i=1}^{i=\alpha} a_i \sharp \psi_i \equiv_k c$.
In this case, all $\psi_i \in \mathsf{EML}(\mathrm{REG})$, therefore, $\psi_i \in \mathsf{EML}_1^+(\mathrm{CFL})$ which lets us use the induction hypothesis on them. With this observation, these cases go through exactly as in the original proof.

The current model $\mathcal{M}$ is of double exponential size in $|\varphi|$ and it is easy to show that it is N-bounded. □

Lemma 5.7 *If φ is $\mathsf{EML}_1^+(\mathrm{CFL})$ satisfiable in some N-bounded model then for some $X \subseteq \mathrm{cl}(0, \varphi)$, $\mathrm{SAT}(X, 0)$ has an accepting computation with $\varphi \in \ker(X)$.*

Proof. Assume that φ is $\mathsf{EML}_1^+(\mathrm{CFL})$ satisfiable in some N-bounded model $\mathcal{M} = \langle T, R, (<_s)_{s\in T}, l\rangle$. So there is $s \in T$ such that $\mathcal{M}, s \models \varphi$ and $\langle \mathcal{M}, s\rangle$ is N-bounded.

Given a formula φ and an $\mathsf{EML}_1^+(\mathrm{CFL})$ model $\mathcal{M}'$ we use $X[s', d]$ with $s' \in T'$ and $d \in [0, |\varphi|]$ to denote the set $\{\psi \in \mathrm{cl}(d, \varphi) : \mathcal{M}', s' \models \psi\}$. We shall show that whenever $\langle \mathcal{M}', s'\rangle$ is N-bounded then $\mathrm{SAT}(X[s', d], d)$ has an accepting computation. We recall that $X[s', d]$ is d-locally consistent. Consequently, we get that $\mathrm{SAT}(X[s, 0], 0)$ has an accepting computation and, by definition, $\varphi \in \ker(X[s, 0])$. The proof is by induction on $d_{max} - d$ where d_{max} is the maximum value such that $\mathrm{cl}(d_{max}, \varphi) \neq \emptyset$.

Base case: $d = d_{max}$.
Any satisfiable set of literals included in $\mathrm{cl}(d_{max}, \varphi)$ is consistent and leads to an accepting computation.

Induction hypothesis: for all $1 \leqslant n \leqslant d' \leqslant |\varphi|$ and $X \subseteq \mathrm{cl}(d', \varphi)$ such that there exist an model $\mathcal{M}' = \langle T', R', (<'_s)_{s\in T'}, l'\rangle$ and $s' \in T'$ with $X_{d'} = \{\psi \in \mathrm{cl}(d', \varphi) : \mathcal{M}', s' \models \psi\}$ and $\langle \mathcal{M}', s'\rangle$ is N-bounded, $\mathrm{SAT}(X[s', d'], d')$ has an accepting computation.

Let $d = n - 1$ and $\mathcal{M}' = \langle T', R', (<'_s)_{s\in T'}, l'\rangle$, with $s' \in T'$ such that $\langle \mathcal{M}', s'\rangle$ is N-bounded. The set $X[s', d]$ is, by definition, d-locally consistent and $\mathsf{EML}_1^+(\mathrm{CFL})$ satisfiable.

For $i \in \{1, \ldots, nb(d+1)\}$, let Y_i be the i^{th} (d+1)-locally consistent set. We write n_i to denote the number of times that the set Y_i holds in the children of s', i.e., $n_i = \mathrm{card}(\{Y_i = X[s'', d+1] : s'' \in T' \text{ and } R'(s', s'')\})$. Observe that, since $\mathcal{M}'$ is N-bounded, $\sum_i n_i \leqslant nb(d+1) \times N$.

This is sufficient to establish that $\mathrm{SAT}(X[s', d], d)$ has an accepting computation. Indeed, the step (consistency) is successful because $X[s', d]$ is d-locally consistent. The guessed number NB is obviously $\sum_i n_i$ and each set Y_i is guessed n_i times in the step (guess-children-from-left-to-right). Additionally, the order in which the sets Y_i are guessed is precisely given by the ordering of the children of the root of $\mathcal{M}'$. Since $\mathcal{M}'$ is a model for $X[s', d]$, for every $i \in [1, nb(d'+1)]$, if $n_i \neq 0$, then the set Y_i is satisfiable in some N-bounded model (namely $\langle \mathcal{M}, s''\rangle$). By the induction hypothesis, $\mathrm{SAT}(Y_i, d+1)$ returns `true`. Each passage to (guess-children-from-left-to-right) as well as the passage to (final-checking) are successful steps because the numbers of children is computed from $\mathcal{M}'$. Hence, $\mathrm{SAT}(X[s', d], d)$ has an accepting computation. □

Theorem 5.8 (Correctness) *A formula φ is $\mathsf{EML}_1^+(\mathrm{CFL})$ satisfiable in some N-bounded model iff for some $X \subseteq \mathrm{cl}(0, \varphi)$, $\mathrm{SAT}(X, 0)$ has an accepting computation and $\varphi \in \ker(X)$.*

We are in position to establish that N is at most doubly exponential in $|\varphi|$.

Lemma 5.9 *There is a polynomial $q(\cdot)$ such that for every formula φ, φ is $\mathsf{EML}_1^+(\mathrm{CFL})$ satisfiable iff φ is satisfiable in some $2^{2^{q(|\varphi|)}}$-bounded model.*

Proof. The proof extends that in [13, Lemma 7]. Only the main change is explained below. Suppose that $\mathcal{C}, \mathcal{B}_1, \ldots, \mathcal{B}_\ell, \neg\mathcal{B}'_1, \ldots, \neg\mathcal{B}'_m$ are the automata-based formulae (or their negation) occurring in some set $\ker(X) \subseteq X \subseteq cl(d, \varphi)$ ($\mathcal{C}$ is the unique CFL-constraint, if any). First, we build a new PDA $\mathcal{C}^+$, over the alphabet $\Sigma = \{Y_1, \ldots, Y_{nb(n+1)}\}$ where $Y_1, \ldots, Y_{nb(n+1)}$ are the only $(n+1)$-locally consistent sets. The automata $\mathcal{C}^+$ and $\mathcal{C}$ have the same sets of states, initial states, stack symbols and final states and $q \xrightarrow{Y} q'$ in $\mathcal{C}^+$ iff $q \xrightarrow{\psi} q'$ in $\mathcal{C}$ for some $\psi \in Y$. For each transition, the action on the stack in the new PDA should be the same as in the original one. The FSAs are also converted into new FSA $\mathcal{B}_1^+, \ldots, \mathcal{B}_\ell^+, \mathcal{B}_1^-, \ldots, \mathcal{B}_m^-$ in the same way as before (obviously, without handling the stack). Next we synchronize all the automata $\mathcal{C}^+, \mathcal{B}_1^+, \ldots, \mathcal{B}_\ell^+$ and the *complement* of $\mathcal{B}_1^-, \ldots, \mathcal{B}_m^-$ into a product PDA $\mathcal{C}^*$ over the alphabet Σ. Observe that this can be done because the intersection of all the FSA yields a FSA and the intersection of a PDA and a FSA provides a PDA. The automaton $\mathcal{C}^*$ is such that for every word $\alpha = Y_{j_1} \cdots Y_{j_\alpha} \in \Sigma^*$, $\alpha \in \mathcal{L}(\mathcal{C}^*)$ iff (i) there are $\psi_1 \in Y_{j_1}, \ldots, \psi_\alpha \in Y_{j_\alpha}$ such that $\psi_1 \cdots \psi_\alpha \in \mathcal{L}(\mathcal{C})$; (ii) for all $i \in [1, \ell]$, there are $\psi_1 \in Y_{j_1}, \ldots, \psi_\alpha \in Y_{j_\alpha}$ s.t. $\psi_1 \cdots \psi_\alpha \in \mathcal{L}(\mathcal{B}_i)$; (iii) for all $i \in [1, m]$, there are no $\psi_1 \in Y_{j_1}, \ldots, \psi_\alpha \in Y_{j_\alpha}$ s.t. $\psi_1 \cdots \psi_\alpha \in \mathcal{L}(\mathcal{B}'_i)$.

Observe first that there is a CFG G such that $\mathcal{L}(G) = \mathcal{L}(\mathcal{A}^*)$ and the size of G is polynomial in $|\mathcal{A}^*|$. Moreover, there exists a FSA $\mathcal{A}^*$ such that $\mathcal{L}(\mathcal{A}^*)$ and $\mathcal{L}(G)$ have the same Parikh image (see e.g. [26]) and $|\mathcal{A}^*|$ is exponential in $|G|$, see e.g. [14]. Consequently, there is a FSA $\mathcal{A}^*$ whose size is at most of double exponential in $|\varphi|$, which is one exponential higher than the bound obtained in [13, Lemma 7] for that part of the argument. The rest of the proof follows that of [13, Lemma 7] *except that we carry values with one exponential higher.* □

Note that for CFGs in Chomsky normal form (every CFG can be converted to such form in polynomial time) a lower bound for the size of an equivalent FSA with identical Parikh image is $\Omega(2^n)$, see e.g. [14]. Hence, the reasoning performed in the proof sketch of Lemma 5.9 can be hardly improved. Another technique would be needed for substantial improvement.

Theorem 5.10 *The satisfiability problem for* $\mathsf{EML}_1^+(\mathrm{CFL})$ *is in* 2EXPSPACE.

Proof. By Theorem 5.8 and Lemma 5.9, φ is $\mathsf{EML}_1^+(\mathrm{CFL})$ satisfiable iff $\mathrm{SAT}(X, 0)$ has an accepting computation and $\varphi \in \ker(X)$ with $N = 2^{2^{q(|\varphi|)}}$. By Lemma 5.5, in that case SAT runs in space polynomial in $2^{|\varphi|} + N$, whence the complexity upper bound 2EXPSPACE by Savitch's Theorem. □

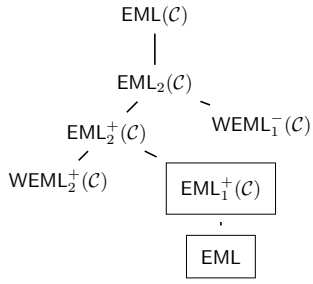
Remark 5.11 The 2EXPSPACE bound can be shown alternatively. Indeed, by Lemma 5.9, φ is satisfiable iff φ is satisfiable by a tree-like model whose depth is bounded by $|\varphi|$ and branching factor is at most $2^{2^{q(|\varphi|)}}$. A nondeterministic 2EXPSPACE algorithm consists in guessing a model of double exponential size and then in performing a model-checking call on it, which can be done in 2EXPSPACE. Savitch's Theorem allows to regain the 2EXPSPACE bound. Note that running SAT consists in guessing the very large model by pieces while

checking satisfaction of subformulae on-the-fly. The optimal complexity lower bound remains unknown.

A natural question about the design of $\mathsf{EML}_1^+(\mathrm{CFL})$ is if we need to simultaneously restrict the positive occurrences of CFL-constraints and the positive occurrences of $\sum_i a_i \sharp \varphi_i > b$ by imposing that either each $a_i > 0$ and $b > 0$ or each $\varphi_i \in \mathsf{EML}(\mathrm{REG})$. Indeed, if we give up this last requirement decidability is lost again. As done in Section 4, we can show that a CFL L over Σ is different from Σ^* iff $\neg\Box \bot \wedge \Box\varphi_{\mathrm{uni}}(p_1, \ldots, p_k) \wedge \sharp \mathrm{L}(p_1, \ldots, p_k) \leqslant 0$ is satisfiable where Σ contains k letters (this formula is not in $\mathsf{EML}_1^+(\mathrm{CFL})$).

6 Conclusion

In this paper, we have investigated the decidability status of EML extensions by adding CFL-constraints, especially those definable from VPLs. For instance, VPLs have been used to express nonregular programs [23] for PDL. Whereas PDL augmented with VPA is decidable, we have shown that the satisfiability problem for $\mathsf{EML}_2(\mathrm{VPL})$ is already undecidable. Also, almost every interesting restriction of this logic still leads to an undecidable fragment. By contrast, in Section 5, we establish that the satisfiability problem for $\mathsf{EML}_1^+(\mathrm{CFL})$ can be solved in 2EXPSPACE by extending proof techniques from [13] that essentially use a Ladner-like algorithm, see e.g. [22], combined with the existence of small solutions for constraint systems. In this fragment, only one CFL-constraint may occur at each modal depth and it should have a positive polarity. Surprisingly, if only one CFL-constraint may occur negatively at each modal depth, then undecidability is back. Schema below contains the decidability status of fragments for $\mathcal{C} \in \{\mathrm{CFL}, \mathrm{VPL}\}$; decidable fragments appear in frames. Our work can be pursued by considering subclasses of VPA such as those definable by semi-simple minded pushdown automata [16]. Such extension are still of interest and our techniques to prove undecidability cannot be applied in that case. Also, the exact characterization of the complexity for the satisfiability problem for $\mathsf{EML}_1^+(\mathrm{CFL})$ is still open.



Acknowledgements: We thank the anonymous referees for their suggestions.

References

[1] Alur, R. and P. Madhusudan, *Visibly pushdown languages*, in: *STOC'04* (2004), pp. 202–211.

[2] Alur, R. and P. Madhusudan, *Adding nesting structure to words*, JACM **56** (2009), pp. 1–43.

[3] Areces, C., G. Hoffmann and A. Denis, *Modal Logics with Counting*, in: *WOLLIC'10*, LNCS **6188** (2010), pp. 98–109.

[4] Barnaba, M. F. and F. D. Caro, *Graded modalities*, Studia Logica **44** (1985), pp. 197–221.
[5] Blackburn, P., M. de Rijke and Y. Venema, "Modal Logic," CUP, 2001.
[6] Bonatti, P., C. Lutz, A. Murano and M. Vardi, *The complexity of enriched mu-calculi*, LMCS **4** (2008).
[7] Borosh, I. and L. Treybig, *Bounds on positive integral solutions of linear diophantine equations*, American Mathematical Society **55** (1976), pp. 299–304.
[8] Cadilhac, M., A. Finkel and P. McKenzie, *On the expressiveness of Parikh automata and related models*, in: *NCMA'11*, books@ocg.at **282** (2011), pp. 103–119.
[9] Calvanese, D. and G. D. Giacomo, *Expressive description logics*, in: *Description Logics Handbook* (2005), pp. 178–218.
[10] Cerrato, C., *General canonical models for graded normal logics*, Studia Logica **49** (1990), pp. 242–252.
[11] Cîrstea, C., A. Kurz, D. Pattinson, L. Schröder and Y. Venema, *Modal logics are coalgebraic*, The Computer Journal **54** (2011), pp. 31–41.
[12] Demri, S. and P. Gastin, "Chapter 'Specification and Verification using Temporal Logics' in Modern Applications of Automata Theory," IIsc Research Monographs, World Scientific, 2011 To appear.
[13] Demri, S. and D. Lugiez, *Complexity of modal logics with Presburger constraints*, JAL **8** (2010), pp. 233–252.
[14] Esparza, J., P. Ganty, S. Kiefer and M. Luttenberger, *Parikh's Theorem: A simple and direct automaton construction*, IPL **111** (2011), pp. 614–619.
[15] Fine, K., *In so many possible worlds*, NDJFL **13** (1972), pp. 516–520.
[16] Harel, D. and M. Kaminsky, *Strengthened Results on Nonregular PDL*, Technical Report MCS99-13, The Weizmann Institute of Science (1999).
[17] Harel, D., A. Pnueli and J. Stavi, *Propositional dynamic logic of nonregular programs*, JCSS **26** (1983), pp. 222–243.
[18] Hollunder, B. and F. Baader, *Qualifying number restrictions in concept languages*, in: *KR'91*, 1991, pp. 335–346.
[19] Ibarra, O., *Restricted one-counter machines with undecidable universe problems*, Mathematical Systems Theory **13** (1979), pp. 181–186.
[20] Kazakov, Y. and I. Pratt-Hartmann, *A note on the complexity of the satisfiability problem for graded modal logic*, in: *LICS'09* (2009), pp. 407–416.
[21] Kupferman, O., U. Sattler and M. Vardi, *The complexity of the graded μ-calculus*, in: *CADE'02*, LNCS **2392** (2002), pp. 423–437.
[22] Ladner, R., *The computational complexity of provability in systems of modal propositional logic*, SIAM Journal of Computing **6** (1977), pp. 467–480.
[23] Löding, C., C. Lutz and O. Serre, *Propositional dynamic logic with recursive programs*, Journal of Logic and Algebric Programming **73** (2007), pp. 51–69.
[24] Mehlhorn, K., *Pebbling mountain ranges and its application to DCFL-recognization*, in: *ICALP'80*, LNCS **85** (1980), pp. 422–435.
[25] Pacuit, E. and S. Salame, *Majority logic*, in: *KR'04* (2004), pp. 598–605.
[26] Parikh, R., *On context-free languages*, JACM **13** (1966), pp. 570–581.
[27] Schröder, L. and D. Pattinson, *PSPACE bounds for rank-1 modal logics*, in: *LICS'06* (2006), pp. 231–240.
[28] Seidl, H., T. Schwentick and A. Muscholl, *Counting in trees*, Texts in Logic and Games **2** (2007), pp. 575–612.
[29] Spaan, E., *The complexity of propositional tense logics*, in: *Diamonds and Defaults* (1993), pp. 287–309.
[30] Tobies, S., *PSPACE reasoning for graded modal logics*, JLC **11** (2001), pp. 85–106.
[31] van der Hoek, W. and J.-J. Meyer, *Graded modalities in epistemic logic*, Logique et Analyse **133–134** (1991), pp. 251–270.
[32] Verma, K. N., H. Seidl and T. Schwentick, *On the complexity of equational Horn clauses*, in: *CADE'05*, LNCS **3632**, 2005, pp. 337–352.
[33] Wolper, P., *Temporal logic can be more expressive*, I & C **56** (1983), pp. 72–99.
[34] Zilio, S. D. and D. Lugiez, *XML schema, tree logic and sheaves automata*, Applicable Algebra in Engineering, Communication and Computing **17** (2006), pp. 337–377.