The Effects of Bounding Syntactic Resources on Presburger LTL

Stéphane Demri, Régis Gascon

The Effects of Bounding Syntactic Resources on Presburger LTL (Extended Abstract)*

Stéphane Demri & Régis Gascon
LSV, ENS Cachan, CNRS, INRIA
{demri,gascon}@lsv.ens-cachan.fr

Abstract

We study decidability and complexity issues for fragments of LTL with Presburger constraints by restricting the syntactic resources of the formulae (the class of constraints, the number of variables and the distance between two states for which counters can be compared) while preserving the strength of the logical operators. We provide a complete picture refining known results from the literature, in some cases pushing forward the known decidability limits. By way of example, we show that model-checking formulae from LTL with quantifier-free Presburger arithmetic over one-counter automata is only PSPACE*-complete. In order to establish the* PSPACE *upper bound, we show that the nonemptiness problem for Büchi one-counter automata taking values in $\mathbb{Z}$ and allowing zero tests and sign tests, is only* NLOGSPACE*-complete.*

1 Introduction

LTL with Presburger constraints. Ubiquity of counter automata in computer science stems from their use as operational models of numerous infinite-state systems, including for instance broadcast protocols [16] and programs with pointer variables [3, 6]. Even the restriction to one counter has found applications in the verification of cryptographic protocols [21] and the validation of XML streams [9]. However, numerous model-checking problems for counter automata, like reachability questions, are known to be undecidable. This does not end the story since many subclasses admit a decidable reachability problem such as reversal-bounded counter machines [19] or flat counter systems [5, 24].

Extending the linear-time temporal logic LTL with Presburger constraints allows to specify quantitative properties about counter systems even though undecidability cannot be avoided (see in [7, 10] decidable fragments by restricting the use of the temporal operators). In this paper, we are interested in the decidability and complexity status of fragments of Presburger LTL restricting the following syntactic resources of the formulae: the class of constraints, the number of counters and the maximal distance between two states for which these counters can be compared. However, we preserve the full strength of the logical operators. Our investigation is based on the standard assumption that restricting the number of variables is a means to define decidable fragments of undecidable logics or to design counter/clock automata with decidable reachability problems, see e.g. [19, 17, 27, 23]. Furthermore this helps to understand the complexity gaps of decidable problems [15, 22]. Our goal is therefore to identify decidable and undecidable fragments of Presburger LTL (both for model-checking and satisfiability problems) refining existing results from [8, 10, 12, 13].

Our contribution. We define $\mathrm{CLTL(DL)}$ as a fragment of Presburger LTL where atomic formulae are difference constraints. The underlying fragment of Presburger arithmetic in $\mathrm{CLTL(DL)}$ is identical to the one in the logic $\mathcal{L}_p$ from [10]. However, it is possible in $\mathrm{CLTL(DL)}$ to state constraints between counters at two non-consecutive states. For instance, "$\mathsf{XX}x = y$" means that the value of y at the current state is equal to the value of x two states further. We call X-length the maximal number of X operators prefixing a counter. As far as undecidability is concerned, we show that satisfiability and model-checking over counter automata for $\mathrm{CLTL(DL)}$ either restricted to formulae of X-length two with at most one counter or to formulae of X-length one with at most two counters are Σ^1_1-complete, improving results from [10, 12]. On the positive side, we prove that model-checking and satisfiability for $\mathrm{CLTL(DL)}$ are PSPACE-complete when restricted to X-length one and to one counter. Hence, we offer a complete and precise taxonomy of $\mathrm{CLTL(DL)}$ fragments with respect to decidability issue.

We follow a standard automata-based approach [31, 20] but we introduce an original symbolic representation of models

*Partially supported by project AVERISS (ANR) and bilateral CNRS/NRF project No19812.

that can be recognized by a fine-tuned class of one-counter automata (instead of standard Büchi automata). A nice property of this method is that it can be generalized to various LTL extensions that define ω-regular classes of models. Among the most technical parts of this work, we show that the nonemptiness problem for this class of counter automata where

- the counter is interpreted in $\mathbb{Z}$,
- there are zero tests and sign tests,
- the accepted language is made of ω-sequences with Büchi acceptance condition,
- the updates of the counter correspond to add one of the values $-1, 0, 1$

is NLOGSPACE-complete. This extends what is known about Büchi automata and variants of one-counter automata [21, 31]. As far as we know, this is a new result obtained by analyzing runs. The details of the full proof are in [14, Sect. 6]. In addition, we show that model-checking LTL with quantifier-free Presburger constraints over one-counter automata is also PSPACE-complete.

Related work. Decidability and complexity issues for LTL variants with Presburger constraints can be found in [7, 10, 12, 13] (see also description logics with concrete domains in [25] and logics of space and time in [2]). Unlike these works, we are studying systematically the effects of bounding the number of variables and the X-length of formulae while preserving the logical operators. This contrasts with fragments shown to be decidable in [7, 10] (not closed under negation).

Model-checking one-counter automata against modal μ-calculus is in PSPACE [28] and more generally model-checking pushdown systems against modal μ-calculus is EXPTIME-complete [33] as well as linear μ-calculus [8, 18]. Herein we show that model-checking linear μ-calculus with quantifier-free Presburger constraints over one-counter automata is PSPACE-complete, refining the above-mentioned works. Satisfiability for this fragment is undecidable as a consequence of [26, Sect. 14.2]. Furthermore, it is worth recalling that even though LTL can be expressed in the modal μ-calculus, these two formalisms have not the same conciseness on common fragment and therefore complexity results cannot always be transferred immediately.
Because of lack of space, the omitted proofs can be found in the preliminary report [14].

2 Temporal logics, automata and Presburger constraints

2.1 From constraint languages to linear-time temporal logics

Constraint languages. Let $\mathrm{VAR} = \{x_0, x_1, \ldots\}$ be a countably infinite set of variables. We consider several fragments of Presburger arithmetic (PA). The difference logic DL is defined by constraints of the form

$$\mathrm{E} ::= x \sim y + d \mid x \sim d \mid \mathrm{E} \wedge \mathrm{E} \mid \neg \mathrm{E}$$

where $x, y \in \mathrm{VAR}$, $d \in \mathbb{Z}$ and $\sim \in \{<, >, \leq, \geq, =\}$. We denote by DL^+ the extension of DL with periodicity constraints of the form either $x \equiv_k c$ or $x \equiv_k y + c$ ($k, c \in \mathbb{N}$). Finally, QFP is the quantifier-free fragment of PA, defined by:

$$\mathrm{E} ::= \sum_{i \in I} a_i x_i \sim d \mid \sum_{i \in I} a_i x_i \equiv_k c \mid \mathrm{E} \wedge \mathrm{E} \mid \neg \mathrm{E}$$

where $a_i \in \mathbb{Z}$ and I is a finite set of indices. Obviously, $\mathrm{DL} \subseteq \mathrm{DL}^+ \subseteq \mathrm{QFP}$. Given a valuation $v : \mathrm{VAR} \to \mathbb{Z}$, the satisfaction relation $v \models \mathrm{E}$ is defined in the obvious way. For instance, $n \equiv_k n'$ iff there is $z \in \mathbb{Z}$ such that $n = n' + kz$. All integers are encoded in binary.

Linear-time temporal logics. Given a constraint language L (typically DL, DL^+ or QFP), we define the logic $\mathrm{CLTL}(L)$ as the extension of LTL where the propositional variables are refined to atomic constraints from L over expressions representing different states of the variables. The formulae of $\mathrm{CLTL}(L)$ are defined by the grammar:

$$\phi ::= \mathrm{E}[x_1 \leftarrow \mathsf{X}^{l_1} x_{j_1}, \ldots, x_n \leftarrow \mathsf{X}^{l_n} x_{j_n}] \mid \phi \wedge \phi \mid$$

$$\neg\phi \mid \mathsf{X}\phi \mid \phi \mathsf{U} \phi$$

where $\mathrm{E}[x_1 \leftarrow \mathsf{X}^{l_1} x_{j_1}, \ldots, x_n \leftarrow \mathsf{X}^{l_n} x_{j_n}]$ is a constraint of L with free variables $x_1, \ldots, x_n$ replaced by terms. A *term* is a variable x_i prefixed by a certain number l of X symbols and is denoted by $\mathsf{X}^l x_i$ (its encoding requires $\mathcal{O}(l + log\ i)$ bits). The symbols X and U are respectively the classical operators "next" and "until" of LTL. We use the notations $\mathsf{F}\phi$ and $\mathsf{G}\phi$ as the abbreviations for $\top \mathsf{U} \phi$ and $\neg \mathsf{F} \neg \phi$. A *one-step constraint* is an atomic formula of the form $\mathrm{E}[x_1 \leftarrow \mathsf{X}^{l_1} x_{j_1}, \ldots, x_n \leftarrow \mathsf{X}^{l_n} x_{j_n}]$ such that $l_1, \ldots, l_n \leq 1$. Given a $\mathrm{CLTL}(L)$ formula ϕ we define its X-*length* $|\phi|_{\mathsf{X}}$ as the maximal number l such that a term of the form $\mathsf{X}^l x$ occurs in ϕ. Intuitively, the X-length defines the size of a frame of consecutive states that can be compared. The models of $\mathrm{CLTL}(L)$ are ω-sequences of valuations $\sigma : \mathbb{N} \to (\mathrm{VAR} \to \mathbb{Z})$ and the satisfaction relation is defined as for LTL except at the atomic level:

- $\sigma, i \models \mathrm{E}[x_1 \leftarrow \mathsf{X}^{l_1} x_{j_1}, ..., x_n \leftarrow \mathsf{X}^{l_n} x_{j_n}]$ iff $(\sigma(i+l_1)(x_{j_1}), ..., \sigma(i+l_n)(x_{j_n})) \models \mathrm{E}$ in PA,
- $\sigma, i \models \phi \wedge \phi'$ iff $\sigma, i \models \phi$ and $\sigma, i \models \phi'$,
- $\sigma, i \models \neg\phi$ iff $\sigma, i \not\models \phi$,
- $\sigma, i \models \mathsf{X}\phi$ iff $\sigma, i+1 \models \phi$,
- $\sigma, i \models \phi \mathsf{U} \phi'$ iff there is $j \geq i$ such that $\sigma, j \models \phi'$ and for every $i \leq k < j$, we have $\sigma, k \models \phi$.

The symbol $\models$, used at the level of the constraint language, is overloaded but this will not lead to any confusion. As usual, a formula $\phi \in \mathrm{CLTL}(L)$ is satisfiable whenever there exists a model σ such that $\sigma, 0 \models \phi$. We write $\mathrm{CLTL}^l_k(L)$ to denote the restriction of $\mathrm{CLTL}(L)$ to formulae with at most k variables and X-length less or equal to l. The satisfiability problem for $\mathrm{CLTL}(\mathrm{QFP})$ can be placed easily in the class Σ^1_1 from the analytical hierarchy.

Constraint automata. A *k-variable L-automaton* $\mathcal{A}$ is a structure $\langle Q, \delta, I, F\rangle$ such that Q is a finite set of states, $I \subseteq Q$ is the set of initial states, $F \subseteq Q$ is the set of final states and $\delta \subseteq Q \times A \times Q$ where A is a finite subset of $1SC_k(L)$, the set of Boolean combinations of one-step constraints from L built over the variables $\{x_1, \ldots, x_k\}$. We use the notation $q \xrightarrow{\mathrm{E}} q'$ as an abbreviation for $\langle q, \mathrm{E}, q'\rangle \in \delta$.

A *configuration* of $\mathcal{A}$ is a tuple $\langle q, \overline{c}\rangle \in Q \times \mathbb{Z}^k$ and we denote by $\overline{c}(i)$ the i^{th} value of $\overline{c}$. The one-step transition relation $\rightarrow$ is defined as follows: $\langle q, \overline{c}\rangle \rightarrow \langle q', \overline{c'}\rangle \stackrel{\text{def}}{\Leftrightarrow}$ there exists $\langle q, \mathrm{E}, q'\rangle \in \delta$ such that if each x_i takes the value $\overline{c}(i)$ and each $\mathsf{X}x_i$ takes the value $\overline{c'}(i)$, then E holds true. We write $\langle q, \overline{c}\rangle \xrightarrow{\mathrm{E}} \langle q', \overline{c'}\rangle$ whenever we need to consider the constraint E on the transition. A finite (resp. infinite) *path* w is a sequence of the form $\{0, \ldots, n\} \rightarrow (Q \times \mathbb{Z}^k)$ (resp. $\mathbb{N} \rightarrow (Q \times \mathbb{Z}^k)$) such that for every $i \in \{0, \ldots, n-1\}$ (resp. for every $i \in \mathbb{N}$) we have $w(i) \rightarrow w(i+1)$. We note $\langle q, \overline{c}\rangle \rightarrow^* \langle q', \overline{c'}\rangle$ if there is a finite path from $\langle q, \overline{c}\rangle$ to $\langle q', \overline{c'}\rangle$. An accepting run for $\mathcal{A}$ is an infinite path w such that $w(0) \in I \times \mathbb{Z}^k$ and the set $\{i \in \mathbb{N} : w(i) \in F \times \mathbb{Z}^k\}$ is infinite (standard Büchi acceptance condition). We write $\mathrm{L}^{\mathrm{symb}}(\mathcal{A})$ to denote the set of ω-words accepted by $\mathcal{A}$ viewed as an automaton over the alphabet A. A $\mathrm{CLTL}(L)$ model σ *realizes* an ω-word $\mathrm{E}_0\mathrm{E}_1 \cdots$ over A iff for every $i \geq 0$, we have $\sigma, i \models \mathrm{E}_i$.

Model-checking. The *model-checking problem* for $\mathrm{CLTL}(L)$ takes as inputs a $\mathrm{CLTL}(L)$ formula ϕ and an L-automaton $\mathcal{A}$ and checks whether there is a $\mathrm{CLTL}(L)$ model σ that realizes some word of $\mathrm{L}^{\mathrm{symb}}(\mathcal{A})$ and such that $\sigma, 0 \models \phi$ (we write $\mathcal{A} \models \phi$). For the restriction to $\mathrm{CLTL}^l_k(L)$, ϕ is in $\mathrm{CLTL}^l_k(L)$ and $\mathcal{A}$ is a k-variable L-automaton. We present the existential version of the problem to simplify forthcoming developments since we also deal with satisfiability but results about the universal version can be withdrawn from those presented herein.

In the rest of the paper, we mainly consider DL^+-automata or subclasses that can simulate non-deterministic Minsky machines. We introduce below subclasses of DL-automata on which we will restrict in some places the model-checking problem.

Counter automata. A k-$\mathbb{Z}$-counter automaton is a restricted DL-automaton such that each transition is of the form $q \xrightarrow{\mathrm{E}} q'$ where E is a conjunction

$$\bigwedge_{i \in \{1 \ldots k\}} \mathrm{E}_{test^i} \wedge \bigwedge_{i \in \{1 \ldots k\}} \mathrm{E}_{update^i}$$

with

- $\mathrm{E}_{test^i} \in \{\top\} \cup \{x_i \sim 0 \mid \sim \in \{<, >, =, \neq\}\}$,
- $\mathrm{E}_{update^i} \in \{\mathsf{X}x_i = x_i + u \mid u \in \mathbb{Z}\}$

for every i. Moreover, we require that the initial values of the counters are equal to zero (with a zero test on every transition from an initial state). For ease of presentation, the elements of $\{\top\} \cup \{x_i \sim 0 \mid \sim \in \{<, >, =, \neq\}\}$ are encoded by $\{\top, <, >, =, \neq\}$, the elements of $\{\mathsf{X}x_i = x_i + u \mid u \in \mathbb{Z}\}$ by $\mathbb{Z}$ and we order the constraints according to an arbitrary ordering of the variables. For instance, the transition

$$q \xrightarrow{\top \wedge x_2 = 0 \wedge \mathsf{X}x_1 = x_1 \wedge \mathsf{X}x_2 = x_2 - 1} q'$$

is encoded by

$$q \xrightarrow{\top, =, 0, -1} q'.$$

A k-$\mathbb{N}$-counter automaton is defined similarly except that we only consider non negative values for the counters. Obviously, one-$\mathbb{Z}$-counter automata with updates in $\mathbb{Z}$ form a proper subclass of one-variable DL-automata that admit also constraints of the form $\mathsf{X}x > x$ or $\mathsf{X}x < x + d$.

In Sect. 3, we define an automata-based approach which differs from [31] by the use of one-$\mathbb{Z}$-counter automata where the updates are restricted to $\{-1, 0, 1\}$, instead of classical Büchi automata. Such automata are called *simple*. Hence, counter automata are used as operational models (inputs of the model-checking problem) and also as language acceptors for adapting the automata-based approach from [31]. Proving the existence of accepting runs for simple one-$\mathbb{Z}$-counter automata is not immediate since we are dealing with Büchi acceptance condition, the counter is interpreted in $\mathbb{Z}$ and zero/sign tests are allowed.

2.2 Improving undecidability boundaries

Satisfiability for $\mathrm{CLTL}(\mathrm{DL})$ is undecidable since we can easily encode the executions of a Minsky machine with

a $\mathrm{CLTL}(\mathrm{DL})$ formula. The proof of [10] provides that $\mathrm{CLTL}_3^1(\mathrm{DL})$ satisfiability is already Σ_1^1-hard. We considerably refine this result by showing that one variable and X-length two or two variables and X-length one is enough for high undecidability.

Theorem 1 *Satisfiability for* $\mathrm{CLTL}_1^2(\mathrm{DL})$ *and* $\mathrm{CLTL}_2^1(\mathrm{DL})$ *are* Σ_1^1*-complete.*

We refer to [14, Section 3.2] for the full proof that contains a reduction from the recurrence problem for nondeterministic Minsky machines, known to be Σ_1^1-complete [1, Lemma 8]. Theorem 1 also refines the undecidability of $\mathrm{CLTL}_2^\omega(\mathrm{DL})$ shown in [12].

We present below the proof for the Σ_1^1-hardness of $\mathrm{CLTL}_1^2(\mathrm{DL})$ satisfiability. The proof for $\mathrm{CLTL}_2^1(\mathrm{DL})$ is obtained by a reduction from $\mathrm{CLTL}_1^2(\mathrm{DL})$ satisfiability (see [14]).

Proof. (satisfiability for $\mathrm{CLTL}_1^2(\mathrm{DL})$ is Σ_1^1-hard)
We show that the existence of an accepting run for two-$\mathbb{N}$-counter automata can be reduced to a satisfiability question in $\mathrm{CLTL}_1^2(\mathrm{DL})$.

First we show that for every two-$\mathbb{N}$-counter automaton $\mathcal{A}$, there is an equivalent two-$\mathbb{N}$-counter automaton $\mathcal{A}'$ computable in logarithmic space in $|\mathcal{A}|$ such that each transition of $\mathcal{A}'$ changes at least the value of one counter. Given a two-$\mathbb{N}$-counter automaton $\mathcal{A} = \langle Q, \delta, I, F\rangle$, $\mathcal{A}' = \langle Q', \delta', I', F'\rangle$ is defined by:

- $Q' \stackrel{\text{def}}{=} Q \cup \{q_t : t \in \delta \text{ and } t = q \xrightarrow{test^1, test^2, =, =} q'\}$, $I' \stackrel{\text{def}}{=} I$ and $F' \stackrel{\text{def}}{=} F$,
- δ' is defined from δ by replacing each transition $t = q \xrightarrow{test^1, test^2, =, =} q' \in \delta$ by $q \xrightarrow{test^1, test^2, +1, =} q_t$ and $q_t \xrightarrow{\top, \top, -1, =} q'$.

One can verify that $\mathcal{A}$ has an accepting run iff $\mathcal{A}'$ has an accepting run.

Now let $\mathcal{A} = \langle Q, \delta, I, F\rangle$ be a two-$\mathbb{N}$-counter automaton such that at each transition at least one counter changes its value. We pose $Q = \{q_1, \ldots, q_n\}$, $I = \{q_{\alpha_1}, \ldots, q_{\alpha_m}\}$ and $F = \{q_{\beta_1}, \ldots, q_{\beta_{m'}}\}$. A configuration of $\langle q_i, c_1, c_2\rangle$ is encoded by a sequence of $2i$ states $c_1, c_1 + c_2 + 1, \ldots, c_1, c_1 + c_2 + 1$ by repeating i times the pair $c_1, c_1 + c_2 + 1$. We recall that a $\mathrm{CLTL}_1^2(\mathrm{DL})$ model is simply an ω-sequence of integers. A new configuration is detected when four consecutive values c, d, c', d' are such that either $c \neq c'$ or $d \neq d'$.

- Let ϕ_{ch} be the formula stating a change of configuration: $\phi_{ch} = x < \mathsf{X}x \ \wedge \ (x \neq \mathsf{X}^2 x \ \vee \ \mathsf{X}(x \neq \mathsf{X}^2 x))$.
- Before_i states that we are just before the configuration with control state q_i:
  $\mathrm{Before}_i \stackrel{\text{def}}{=} \phi_{ch} \ \wedge \ \mathsf{X}^2(\bigwedge_{0 \leq j < i-1} \mathsf{X}^{2j}(x = \mathsf{X}^2 x \ \wedge \ \mathsf{X}(x = \mathsf{X}^2 x)) \ \wedge \ \mathsf{X}^{2(i-1)}\phi_{ch}$.
- Initial configuration:
  $\phi_{init} \stackrel{\text{def}}{=} x_1 = 0 \ \wedge \ x_2 = 1 \ \wedge \ \bigvee_{1 \leq i \leq m}(\bigwedge_{0 \leq j < \alpha_i - 1} \mathsf{X}^{2j}(x = \mathsf{X}^2 x \wedge \mathsf{X}(x = \mathsf{X}^2 x)) \wedge \bigvee_{\langle q_{\alpha_i}, \phi, q_{j'}\rangle \in \delta} \mathsf{X}^{2(\alpha_i - 1)}(\phi' \ \wedge \ \mathrm{Before}_{j'}))$.
  ϕ' is defined below.
- Recurring elements of F:
  $\phi_{rec} \stackrel{\text{def}}{=} \bigvee_{1 \leq i \leq m'} \mathsf{GF}\mathrm{Before}_{\beta_i}$.
- Simulation of the run:
  $\phi_{run} \stackrel{\text{def}}{=} \mathsf{G} \bigwedge_{1 \leq i \leq n}(\mathrm{Before}_i \Rightarrow \bigvee_{\langle q_i, \phi, q_j\rangle \in \delta} \mathsf{X}^{2i}(\phi' \ \wedge \ \mathrm{Before}_j))$ where ϕ' is obtained from ϕ by replacing
  - $x_1 = 0$ by $x = 0$,
  - $x_2 = 0$ by $\mathsf{X}x = x + 1$,
  - $\mathsf{X}x_1 = x_1 + d_1$ by $\mathsf{X}^2 x = x + d_1$ for every $d_1 \in \{-1, 0, 1\}$ and,
  - $\mathsf{X}x_2 = x_2 + d_2$ by $\bigwedge_{d_1 \in \{-1,0,1\}}(\mathsf{X}^2 x = x + d_1) \Rightarrow \mathsf{X}(\mathsf{X}^2 x = x + (d_1 + d_2))$ for every $d_2 \in \{-1, 0, 1\}$.

The automaton $\mathcal{A}$ has an accepting run iff $\phi_{init} \wedge \phi_{run} \wedge \phi_{rec}$ is satisfiable. □

Moreover, the satisfiability problem can be reduced to the model-checking problem since $\phi \in \mathrm{CLTL}(\mathrm{DL})$ is satisfiable iff $\mathcal{A}_\top \models \phi$ where $\mathcal{A}_\top$ is the DL-automaton that accepts all the executions (DL-automata are more liberal than counter automata).

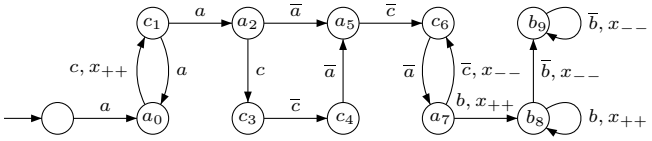
Corollary 1 *The model-checking problems for* $\mathrm{CLTL}_2^1(\mathrm{DL})$ *and* $\mathrm{CLTL}_1^2(\mathrm{DL})$ *are* Σ_1^1*-complete.*

The Σ_1^1 upper bound is obtained by reducing model-checking to satisfiability for $\mathrm{CLTL}(\mathrm{DL})$ along the lines of [29]. In order to simulate a proposition p, we use a constraint of the form $x = 0$ assuming that x is not used already for other purposes. By close inspection of the proof of Theorem 1, one can also show that satisfiability and model-checking for $\mathrm{CLTL}_2^1(\mathrm{DL})$ and $\mathrm{CLTL}_1^2(\mathrm{DL})$ but restricted to the sometime operator F (instead of until) are also Σ_1^1-hard.

3 PSPACE-completeness of $\mathrm{CLTL}_1^1(\mathrm{DL}^+)$ with propositional variables

To complete the results of Section 2, we show that satisfiability for $\mathrm{CLTL}_1^1(\mathrm{DL})$ and model-checking $\mathrm{CLTL}_1^1(\mathrm{DL})$ formulae over 1-variable DL-automata are PSPACE-complete. To do so, we establish a PSPACE upper bound for satisfiability of the richer logic $\mathrm{CLTL}_1^1(\mathrm{DL}^+, \mathrm{PROP})$, including periodicity constraints of the form $x \equiv_k y + c$, $x \equiv_k c$ $(k, c \in \mathbb{N})$ and propositional variables that can be viewed as specific variables with a strict discipline on their constraints. As a matter of fact, the above-mentioned

results not only complete our classification but also many problems on one-counter automata/nets can be encoded in $\mathrm{CLTL}(\mathrm{DL}^+, \mathrm{PROP})$. These problems come from several applications: verification of cryptographic protocols [21], validation of XML streams (string representations of XML documents) [9], or resolution of the identification problem [32]. By way of example, the class of one-variable DL-automata properly contains the one-counter automata that are used to validate XML streams against a recursive DTD in [9, Sect.5]. Below is the one-counter automaton recognizing the language $\{(ac)^n a(\varepsilon|c\overline{c})\overline{a}(\overline{ca})^n b^m \overline{b}^m \,:\, n, m \geq 1\}$ (we omit the tests related to zero):



where x is a counter and the alphabet is partitioned into a set of opening tags $\{a, b, c\}$ and a set of corresponding closing tags $\{\overline{a}, \overline{b}, \overline{c}\}$. The problem of checking whether a given word belongs to this language, a key problem in [9, Sect.5], can be expressed in our formalism. $\mathrm{CLTL}_1^1(\mathrm{DL})$ can also express concisely richer standard properties, for instance non-trivial safety properties of the form $\mathsf{G}(x < 2^m)$ or liveness properties such that $\mathsf{G}(x \equiv_{2^n} 0 \Rightarrow \mathsf{F}(x \equiv_{3^m} 1))$.

We dedicate the remaining of this section to prove decidability of the satisfiability problem for $\mathrm{CLTL}_1^1(\mathrm{DL}^+, \mathrm{PROP})$, which is partially based on the abstraction of models. Without any loss of generality, we can assume that all the atomic formulae involving both x and $\mathsf{X}x$ are of the form $\mathsf{X}x \sim x + d$ and $\mathsf{X}x \equiv_k x + c$ ($d \in \mathbb{Z}$ and $k, c \in \mathbb{N}$).

3.1 Symbolic models

Let $\mathrm{PROP} = \{p_1, p_2, \ldots\}$ be a countably infinite set of propositional variables. We define the logic $\mathrm{CLTL}(\mathrm{DL}^+, \mathrm{PROP})$ as the extension of $\mathrm{CLTL}(\mathrm{DL}^+)$ by adding propositional variables at the atomic level. The main reason for introducing propositional variables stems from the fact that then satisfiability subsumes model-checking. Models of $\mathrm{CLTL}(\mathrm{DL}^+, \mathrm{PROP})$ are pairs $\langle \sigma_1, \sigma_2 \rangle$ such that $\sigma_1 : \mathbb{N} \to 2^{\mathrm{PROP}}$ is a standard LTL model and $\sigma_2 : \mathbb{N} \to (\mathrm{VAR} \to \mathbb{Z})$ is a $\mathrm{CLTL}(\mathrm{DL}^+)$ model. The satisfaction relation is defined as for $\mathrm{CLTL}(\mathrm{DL})$ except at the atomic level: $\langle \sigma_1, \sigma_2 \rangle, i \models p \stackrel{\text{def}}{\Leftrightarrow} p \in \sigma_1(i)$ and given an atomic formula E based on DL^+, $\langle \sigma_1, \sigma_2 \rangle, i \models \mathrm{E} \stackrel{\text{def}}{\Leftrightarrow} \sigma_2, i \models \mathrm{E}$ in $\mathrm{CLTL}(\mathrm{DL}^+)$. There is no restriction on propositional variables in each fragment $\mathrm{CLTL}_k^l(\mathrm{DL}^+, \mathrm{PROP})$.

In order to build an automaton that recognizes the symbolic models of a $\mathrm{CLTL}_1^1(\mathrm{DL}^+)$ formula, we introduce below a symbolic representation of valuations. Let X be a finite set of one-step constraints from DL^+ built over the variable x. We consider the following syntactic resources of X.

- $\mathrm{CONS}_x = \{d_{\min}, \ldots, d_{-1}, d_0, d_1, \ldots, d_{\max}\}$ is the set of constants occurring in X in constraints of the form either $x \sim d$ or $\mathsf{X}x \sim d$. We suppose that $d_{\min} < \cdots < d_{-1} < d_0 < d_1 < \cdots < d_{\max}$.
- $\mathrm{CONS}_{\mathrm{step}} = \{e_{\min'}, \ldots, e_{-1}, e_0, e_1, \ldots, e_{\max'}\}$ is the set of constants occurring in X in constraints of the form $\mathsf{X}x \sim x + e$. We suppose that $e_{\min'} < \cdots < e_{-1} < e_0 < e_1 < \cdots < e_{\max'}$.
- K is the least common multiple of the integers k such that $\equiv_k$ occurs in X.

Wlog, we can assume that $d_0 = e_0 = 0$, $d_{\max} \geq 0$, $e_{\max'} \geq 0$, $d_{\min} \leq 0$ and $e_{\min'} \leq 0$.

We define an abstraction of valuations like regions for timed automata and we shall prove that this abstraction fits exactly our goal. A map $\{x, \mathsf{X}x\} \to \mathbb{Z}$ (also viewed as a pair $\langle z_1, z_2 \rangle \in \mathbb{Z}^2$) is represented by a tuple $sv = \langle \mathrm{E}_x, \mathrm{E}_m, \mathrm{E}'_x, \mathrm{E}'_m, \mathrm{E}_s \rangle \in \mathrm{C}_x \times \mathrm{Mod}_x \times \mathrm{C}_{\mathsf{X}x} \times \mathrm{Mod}_{\mathsf{X}x} \times \mathrm{C}_{\mathrm{step}}$ (depending on X) such that for each term $t \in \{x, \mathsf{X}x\}$,

- C_t is composed of constraints of the form below
  - $(d_i < t) \wedge (t < d_{i+1})$ for $i \in \{\min, .., \max - 1\}$,
  - $t = d_i$ for $i \in \{\min, \ldots, \max\}$,
  - $t < d_{\min}$ and $d_{\max} < t$,
- Mod_t is composed of the constraints $t \equiv_K c$ for $c \in \{0, \ldots, K-1\}$,
- $\mathrm{C}_{\mathrm{step}}$ is composed of constraints of the form
  - $x + e_i < \mathsf{X}x \wedge \mathsf{X}x < x + e_{i+1}$ for $i \in \{\min', \ldots, \max' - 1\}$,
  - $\mathsf{X}x = x + e_i$ for $i \in \{\min', \ldots, \max'\}$,
  - $\mathsf{X}x < x + e_{\min'}$ and $x + e_{\max'} < \mathsf{X}x$.

We call such a tuple a *symbolic valuation* and we write $\mathrm{SV}(X)$ to denote the set of symbolic valuations w.r.t. X. Given a $\mathrm{CLTL}_1^1(\mathrm{DL}^+, \mathrm{PROP})$ formula ϕ, $\mathrm{SV}(\phi)$ is the set of symbolic valuations w.r.t. the set of atomic constraints occurring in ϕ. The size of $\mathrm{SV}(\phi)$ is exponential in the size of ϕ and each element sv of $\mathrm{SV}(\phi)$ can be encoded in polynomial space in the size of ϕ. By definition, we write $v \models sv$ if the valuation v satisfies all the constraints in the symbolic valuation sv.

Lemma 1 *Let X be a finite set of one-step constraints from $\mathrm{CLTL}_1^1(\mathrm{DL}^+)$ built over the variable x.*
(I) *For every map $v : \{x, \mathsf{X}x\} \to \mathbb{Z}$ there is a unique symbolic valuation in $SV(X)$ denoted by $sv(v)$ such that*

$v \models sv(v)$.
(II) *For all the maps* $v, v' : \{x, \mathsf{X}x\} \to \mathbb{Z}$ *such that* $sv(v) = sv(v')$ *and for every* $\mathrm{E} \in X$, $v \models \mathrm{E}$ *iff* $v' \models \mathrm{E}$.

Proof. **(I)** Given $sv \in \mathrm{SV}(X)$, let V_{sv} be the set of pairs $\langle z_1, z_2 \rangle \in \mathbb{Z}^2$ such that $\langle z_1, z_2 \rangle \models sv$. It is easy to show that $\{V_{sv} : sv \in \mathrm{SV}(X), V_{sv} \neq \emptyset\}$ is a partition of $\mathbb{Z}^2$.
(II) Let v and v' be two valuations such that $sv(v) = sv(v') = \langle \mathrm{E}_x, \mathrm{E}'_x, \mathrm{E}_m, \mathrm{E}'_m, \mathrm{E}_s \rangle$ and suppose that $v \models \mathrm{E}$. We proceed by induction on the structure of E.

- If E is of the form $x = d$ then E_x must be equal to E because $d \in \mathrm{CONS}_x$ (and $v \models \mathrm{E}_x$). Since v' also satisfies E_x, we have $v' \models \mathrm{E}$.
- If E is of the form $x < d$ then, (since $v \models \mathrm{E}_x$), E_x must be equal either to $x = d'$ with $d' < d$ or to $d'' < x \wedge x < d'$ with $d' \leq d$. Since $v' \models \mathrm{E}_x$, we have $v' \models \mathrm{E}$.
- When E is of the form $\mathsf{X}x \sim d$ (resp. $\mathsf{X}x \sim x + d$), the proof is similar, using the constraint E'_x (resp. E_s).
- Let E be of the form $x \equiv_k c$. We consider the constraint E_m of the form $x \equiv_K c'$. By definition, k divides K and so E_m implies $x \equiv_k c'_r$ where c'_r is the remainder of the division of c' by k. As c and c'_r belong to $\{0, \ldots, k-1\}$ and v satisfies both E and E_m, c must be equal to c'_r. Since $v' \models \mathrm{E}_m$ and E_m implies E, we have $v' \models \mathrm{E}$.
- When E is of the form $\mathsf{X}x \equiv_k c$ (resp. $\mathsf{X}x \equiv_k x + c$) the proof is similar by using the constraint E'_m (resp. $\mathrm{E}_m \wedge \mathrm{E}'_m$).
- Now suppose that E and E' are satisfied by v iff they are satisfied by v'.
  - If $v \models \mathrm{E} \wedge \mathrm{E}'$ then $v \models \mathrm{E}$ and $v \models \mathrm{E}'$. By the induction hypothesis $v' \models \mathrm{E}$ and $v' \models \mathrm{E}'$ whence $v' \models \mathrm{E} \wedge \mathrm{E}'$.
  - If $v \models \neg \mathrm{E}$ then $v \not\models \mathrm{E}$. Using the induction hypothesis, we have $v \not\models \mathrm{E}'$ and we conclude that $v' \models \neg \mathrm{E}$.

□

Given an atomic formula E from $\mathrm{CLTL}^1_1(\mathrm{DL}^+)$, we note $sv \models_{\mathrm{symb}} \mathrm{E}$ iff for every valuation v such that $sv(v) = sv$ we have $v \models \mathrm{E}$. A sequence of symbolic valuations w.r.t. ϕ is a word $\rho : \mathbb{N} \to \mathrm{SV}(\phi)$ and ρ is satisfiable iff there is a model $\sigma : \mathbb{N} \to \mathbb{Z}$ for $\mathrm{CLTL}^1_1(\mathrm{DL}^+)$ such that for all $i \in \mathbb{N}$, we have $\sigma, i \models \rho(i)$ (we write $\sigma \models \rho$). A *symbolic model* w.r.t. ϕ is a pair $\langle \sigma_1, \rho \rangle$ such that $\sigma_1 : \mathbb{N} \to 2^{\mathrm{PROP}}$ and $\rho : \mathbb{N} \to \mathrm{SV}(\phi)$. The symbolic satisfaction relation $\models_{\mathrm{symb}}$ is extended to symbolic models. The definition is identical to the satisfaction relation of $\mathrm{CLTL}(\mathrm{DL}^+, \mathrm{PROP})$ except for atomic constraints: $\langle \sigma, \rho \rangle, i \models_{\mathrm{symb}} \mathrm{E} \stackrel{\text{def}}{\Leftrightarrow} \rho(i) \models_{\mathrm{symb}} \mathrm{E}$.

3.2 Automata-based approach

We show in the following that given a formula ϕ in $\mathrm{CLTL}^1_1(\mathrm{DL}^+, \mathrm{PROP})$ we can build an automaton $\mathcal{A}_\phi$ recognizing symbolic representations of the models satisfying ϕ. In order to define $\mathcal{A}_\phi$, we slightly extend the transitions of simple one-$\mathbb{Z}$-counter automata (with updates in $\{-1, 0, 1\}$) by decorating them with elements from $\Sigma \cup \{\varepsilon\}$ where Σ is a finite alphabet and with the identical conditions about the constraints authorized in one-$\mathbb{Z}$-counter automata. Since adding an alphabet is motivated by the need to consider the automata as language acceptors we define $\mathrm{L}'(\mathcal{A}) = \{\sigma : \mathbb{N} \to (\Sigma \cup \{\varepsilon\}) \mid \text{there is an accepting run } w \text{ such that } \forall i \ w(i) \xrightarrow{\sigma(i), \mathrm{E}} w(i+1)\}$ and $\mathrm{L}(\mathcal{A}) = \{\sigma^{\setminus \varepsilon} \mid \sigma \in \mathrm{L}'(\mathcal{A}),$ elements of Σ occur infinitely often in $\sigma\}$ where $\sigma^{\setminus \varepsilon}$ is obtained from σ by erasing all ε. $\mathrm{L}(\mathcal{A})$ is the language accepted by $\mathcal{A}$. The construction of $\mathcal{A}_\phi$ relies on the following observation.

Lemma 2 *A* $\mathrm{CLTL}^1_1(\mathrm{DL}^+, \mathrm{PROP})$ *formula* ϕ *is satisfiable iff there exist a symbolic model* $\langle \sigma_1, \rho \rangle$ *such that* $\langle \sigma_1, \rho \rangle \models_{\mathrm{symb}} \phi$ *and a* $\mathrm{CLTL}^1_1(\mathrm{DL}^+)$ *model* σ_2 *such that* $\sigma_2 \models \rho$.

Thus, $\mathcal{A}_\phi$ is defined as the intersection of two automata $\mathcal{A}_{\mathrm{symb}}$ and $\mathcal{A}_{\mathrm{sat}}$ such that $\mathrm{L}(\mathcal{A}_{\mathrm{symb}})$ is the set of symbolic models that symbolically satisfies ϕ and $\mathrm{L}(\mathcal{A}_{\mathrm{sat}})$ is the set of symbolic models $\langle \sigma_1, \rho \rangle$ such that ρ is satisfiable. Both automata are simple one-$\mathbb{Z}$-counter automata over the alphabet $\Sigma = (2^{\mathrm{PROP}} \times \mathrm{SV}(\phi))$ and $\mathcal{A}_{\mathrm{symb}}$ is essentially a finite-state automaton without counters. The automaton $\mathcal{A}_{\mathrm{symb}}$ is built as in [31] for LTL except at the atomic level. We define $cl(\phi)$ the closure of ϕ with a slight modification to consider both atomic constraints and propositional variables and an atom of ϕ is a maximally consistent subset of $cl(\phi)$. Let $\mathcal{A}'_{\mathrm{symb}}$ be the generalized Büchi automaton defined by the structure $\langle Q, \delta, I, F \rangle$ such that:

- Q is the set of atoms of ϕ,
- $I = \{X \in Q : \phi \in X\}$,
- $X \xrightarrow{\langle P, sv \rangle, \top, 0} Y$ iff
  (atomic) $P = X \cap \mathrm{PROP}$ and
  for every atomic E in X, $sv \models_{\mathrm{symb}} \mathrm{E}$,
  (1-step) for every $\mathsf{X}\psi \in cl(\phi)$, $\mathsf{X}\psi \in X$ iff $\psi \in Y$,
- Let $\{\psi_1 \mathsf{U} \phi_1, \ldots, \psi_n \mathsf{U} \phi_n\}$ be the set of until formulae in $cl(\phi)$. We pose $F = \{F_1, \ldots, F_n\}$ where $F_i = \{X \in Q : \psi_i \mathsf{U} \phi_i \notin X \text{ or } \phi_i \in X\}$ for every $i \in \{1, \ldots, n\}$.

The automaton $\mathcal{A}_{\mathrm{symb}}$ is the (non generalized) Büchi automaton equivalent to $\mathcal{A}'_{\mathrm{symb}}$ which can be built in logarithmic space in the size of $\mathcal{A}'_{\mathrm{symb}}$.

The automaton $\mathcal{A}_\phi$ is obtained by synchronizing $\mathcal{A}_{\mathrm{symb}}$ and $\mathcal{A}_{\mathrm{sat}}$. Let us pose $\mathcal{A}_{\mathrm{symb}} = \langle Q_{sy}, \delta_{sy}, I_{sy}, F_{sy} \rangle$

and $\mathcal{A}_{\rm sat} = \langle Q_{sa}, \delta_{sa}, I_{sa}, F_{sa}\rangle$. The automaton $\mathcal{A}_\phi = \langle Q, \delta, I, F\rangle$ is defined by:

- $Q = Q_{sy} \times Q_{sa}$, $I = I_{sy} \times I_{sa}$, $F = F_{sy} \times Q_{sa}$ (we will have $Q_{sa} = F_{sa}$),
- $\langle q_1, q_2\rangle \xrightarrow{\varepsilon,t,u} \langle q_1', q_2'\rangle \in \delta \stackrel{\text{def}}{\Leftrightarrow} q_1 = q_1'$ and $q_2 \xrightarrow{\varepsilon,t,u} q_2' \in \delta_{sa}$,
- $\langle q_1, q_2\rangle \xrightarrow{\langle P,sv\rangle,t,u} \langle q_1', q_2'\rangle \in \delta \stackrel{\text{def}}{\Leftrightarrow}$ $q_1 \xrightarrow{\langle P,sv\rangle,\top,0} q_1' \in \delta_{sy}$ and $q_2 \xrightarrow{\langle P,sv\rangle,t,u} q_2' \in \delta_{sa}$.

Lemma 3 *Given a formula ϕ, one can build a simple one-$\mathbb{Z}$-counter automaton $\mathcal{A}_{\rm sat}$ with alphabet $\Sigma = 2^{\rm PROP} \times {\rm SV}(\phi)$ such that ${\rm L}(\mathcal{A}_{\rm sat})$ is the set of satisfiable symbolic models w.r.t ϕ.*
Moreover, $\mathcal{A}_\phi$ can be effectively built from ϕ in polynomial space thanks to the way $\mathcal{A}_{\rm sat}$ is defined.

Proof. We describe the construction of the automaton $\mathcal{A}_{\rm sat}$ recognizing exactly the set of symbolic models $\langle \sigma_1, \rho\rangle$ such that ρ is satisfiable. We recall that the set ${\rm CONS}_x$ is such that $d_0 = 0$, $d_{\max} \geq 0$ and $d_{\min} \leq 0$.

The alphabet of $\mathcal{A}_{\rm sat}$ is $2^{\rm PROP} \times {\rm SV}(\phi)$ but since the set of propositional variables is not constrained in $\mathcal{A}_{\rm sat}$, we omit them in the technical developments below. The construction of $\mathcal{A}_{\rm sat}$ is done in a modular fashion. $\mathcal{A}_{\rm sat}$ is made of a network of components/gadgets and it is of exponential size in the size of ϕ. A component is defined as a simple one-$\mathbb{Z}$-counter automaton $\langle \Sigma, Q, \delta, I, F\rangle$ such that

- I and F are singletons,
- δ is a subset of $(Q \setminus F) \times \{\varepsilon\} \times \{\top, =, \neq, >, <\} \times \{-1, 0, 1\} \times (Q \setminus I)$.

The unique state in I (resp. F) is called the input (resp. output) state of the component. Components are connected in the network by defining transitions between input states and output states. Each component in $\mathcal{A}_{\rm sat}$ has the function either to check a property of the counter from constraints in ${\rm C}_x$ or to update the counter according to constraints in ${\rm Mod}_x$ or ${\rm Mod}_{\mathsf{X}x} \times {\rm C}_{\rm step}$. We define below the components $\mathcal{A}_{{\rm E},sv}$ for some ${\rm E} \in {\rm C}_x \cup {\rm Mod}_x \cup ({\rm Mod}_{\mathsf{X}x} \times {\rm C}_{\rm step})$ and $sv \in {\rm SV}(\phi)$. We write $q_{in}^{{\rm E},sv}$ (resp. $q_{out}^{{\rm E},sv}$) to denote the input (resp. output) state of $\mathcal{A}_{{\rm E},sv}$ (when the context is clear we shortly write q_{in} and q_{out}, respectively). Each component $\mathcal{A}_{{\rm E},sv}$ enforces that the next symbolic valuation that is guessed is precisely sv. For every $sv = \langle {\rm E}_x, {\rm E}_m, {\rm E}_x', {\rm E}_m', {\rm E}_s\rangle \in {\rm SV}(\phi)$, we define the following components:

- $\mathcal{A}_{{\rm E}_x,sv}$ is such that for every $c \in \mathbb{Z}$, $\langle q_{in}^{{\rm E}_x,sv}, c\rangle \rightarrow^* \langle q_{out}^{{\rm E}_x,sv}, c'\rangle$ iff $c = c'$ and $[x \leftarrow c] \models {\rm E}_x$. This component checks that c satisfies ${\rm E}_x$. Fig. 1 contains some graphical representation of components $\mathcal{A}_{x=d_i,sv}$ and $\mathcal{A}_{d_i<x<d_{i+1},sv}$ when $d_i \geq 0$. Components with $d_i \leq 0$ can be defined analogously.

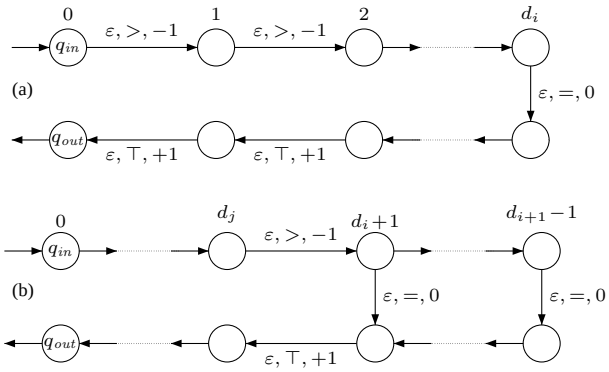


Figure 1. (a) $\mathcal{A}_{x=d_i,sv}$ and (b) $\mathcal{A}_{d_i<x<d_{i+1},sv}$

- $\mathcal{A}_{\langle {\rm E}_m', {\rm E}_s\rangle,sv}$ is such that for every $c \in \mathbb{Z}$, $[x \leftarrow c] \models {\rm E}_m$ and $\langle q_{in}^{\langle {\rm E}_m', {\rm E}_s\rangle,sv}, c\rangle \rightarrow^* \langle q_{out}^{\langle {\rm E}_m', {\rm E}_s\rangle,sv}, c'\rangle$ iff $[x \leftarrow c, \mathsf{X}x \leftarrow c'] \models {\rm E}_m' \wedge {\rm E}_s$. This component updates the counter according to $\langle {\rm E}_m', {\rm E}_s\rangle$. Fig. 2 contains a graphical representation of the component $\mathcal{A}_{\langle {\rm E}_m', {\rm E}_s\rangle,sv}$ with ${\rm E}_m = x \equiv_2 1$ (from sv), ${\rm E}_m' = \mathsf{X}x \equiv_2 0$ and ${\rm E}_s = x < \mathsf{X}x < x + 7$. To build $\mathcal{A}_{\langle {\rm E}_m', {\rm E}_s\rangle,sv}$, we determine on the fly (using ${\rm E}_m$, ${\rm E}_m'$ and ${\rm E}_s$) that $\mathsf{X}x = x + i$ for some $i \in \{1, 3, 5\}$.

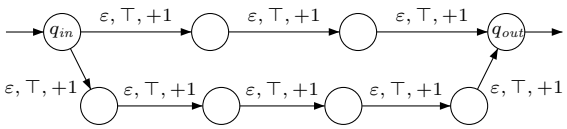


Figure 2. $\mathcal{A}_{\langle {\rm E}_m', {\rm E}_s\rangle,sv}$

- $\mathcal{A}_{{\rm E}_m,sv}$ is such that for every $c \in \mathbb{Z}$, $\langle q_{in}^{{\rm E}_m,sv}, 0\rangle \rightarrow^* \langle q_{out}^{{\rm E}_m,sv}, c\rangle$ iff $[x \leftarrow c] \models {\rm E}_m$. This component updates the counter from 0 to a value satisfying ${\rm E}_m$ (only used at the beginning of the run). Fig. 3 contains a graphical representation of some component $\mathcal{A}_{x \equiv_K c,sv}$.

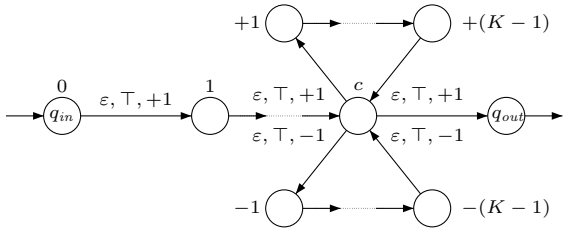


Figure 3. $\mathcal{A}_{x \equiv_K c,sv}$

The automaton $\mathcal{A}_{\rm sat} = \langle \Sigma, Q, \delta, I, F\rangle$ is defined as the "disjoint union" of the above-mentioned components with

an additional initial state s_0, $F = Q$ and with the following additional transitions.

- For every $sv = \langle \mathrm{E}_x, \mathrm{E}_m, \mathrm{E}'_x, \mathrm{E}'_m, \mathrm{E}_s \rangle \in \mathrm{SV}(\phi)$, $s_0 \xrightarrow{\varepsilon,\top,0} q_{in}^{\mathrm{E}_m,sv} \in \delta$. This corresponds to decide which constraint E_m the first value of the counter satisfies. The only way for the run to continue is to enter further in $\mathcal{A}_{\mathrm{E}_m,sv}$.
- For every $sv = \langle \mathrm{E}_x, \mathrm{E}_m, \mathrm{E}'_x, \mathrm{E}'_m, \mathrm{E}_s \rangle \in \mathrm{SV}(\phi)$, $q_{out}^{\mathrm{E}_m,sv} \xrightarrow{\varepsilon,\top,0} q_{in}^{\mathrm{E}_x,sv} \in \delta$. When the control state is $q_{out}^{\mathrm{E}_m,sv}$, the counter satisfies E_m. Now we want to check E_x. So, the only way to continue the run is to enter in $\mathcal{A}_{\mathrm{E}_x,sv}$.
- For every $sv = \langle \mathrm{E}_x, \mathrm{E}_m, \mathrm{E}'_x, \mathrm{E}'_m, \mathrm{E}_s \rangle \in \mathrm{SV}(\phi)$, $q_{out}^{\mathrm{E}_x,sv} \xrightarrow{\varepsilon,\top,0} q_{in}^{\langle \mathrm{E}'_m, \mathrm{E}_s \rangle,sv} \in \delta$. When the control state is $q_{out}^{\mathrm{E}_x,sv}$, this means that the counter satisfies E_x and it is now time to update it according to $\langle \mathrm{E}'_m, \mathrm{E}_s \rangle$. The only way for the run to continue is to enter further in $\mathcal{A}_{\langle \mathrm{E}'_m, \mathrm{E}_s \rangle,sv}$.
- For all $sv_1 = \langle (\mathrm{E}_x)_1, (\mathrm{E}_m)_1, (\mathrm{E}'_x)_1, (\mathrm{E}'_m)_1, (\mathrm{E}_s)_1 \rangle \in \mathrm{SV}(\phi)$ and $sv_2 = \langle (\mathrm{E}_x)_2, (\mathrm{E}_m)_2, (\mathrm{E}'_x)_2, (\mathrm{E}'_m)_2, (\mathrm{E}_s)_2 \rangle \in \mathrm{SV}(\phi)$ s.t. $(\mathrm{E}'_x)_1[\mathsf{X}x \leftarrow x] = (\mathrm{E}_x)_2$ and $(\mathrm{E}'_m)_1[\mathsf{X}x \leftarrow x] = (\mathrm{E}_m)_2$, $q_{out}^{\langle (\mathrm{E}'_m)_1, (\mathrm{E}_s)_1 \rangle, sv_1} \xrightarrow{sv_1,\top,0} q_{in}^{(\mathrm{E}_x)_2, sv_2} \in \delta$. The letter sv_1 can be read since all the verifications have been successful. The only way for the run to continue is to enter further in $\mathcal{A}_{(\mathrm{E}_x)_2,sv_2}$. A new symbolic valuation sv_2 is guessed but sv_2 has to agree with sv_1 on some constraints.

For all symbolic valuations $sv_1 = \langle (\mathrm{E}_x)_1, (\mathrm{E}_m)_1, (\mathrm{E}'_x)_1, (\mathrm{E}'_m)_1, (\mathrm{E}_s)_1 \rangle \in \mathrm{SV}(\phi)$ and $sv_2 = \langle (\mathrm{E}_x)_2, (\mathrm{E}_m)_2, (\mathrm{E}'_x)_2, (\mathrm{E}'_m)_2, (\mathrm{E}_s)_2 \rangle \in \mathrm{SV}(\phi)$, for all $c, c' \in \mathbb{Z}$, the propositions below are equivalent (by construction of the components):

(I) $[x \leftarrow c] \models (\mathrm{E}_m)_1$ and $\langle q_{in}^{(\mathrm{E}_x)_1, sv_1}, c \rangle \xrightarrow{\varepsilon}{}^* \langle q_{in}^{\langle (\mathrm{E}'_m)_1, (\mathrm{E}_s)_1 \rangle, sv_1}, c \rangle \xrightarrow{\varepsilon}{}^* \langle q_{out}^{\langle (\mathrm{E}'_m)_1, (\mathrm{E}_s)_1 \rangle, sv_1}, c' \rangle \xrightarrow{sv_1} \langle q_{in}^{(\mathrm{E}_x)_2, sv_2}, c' \rangle \xrightarrow{\varepsilon}{}^* \langle q_{out}^{(\mathrm{E}_x)_2, sv_2}, c' \rangle$,

(II) $[x \leftarrow c, \mathsf{X}x \leftarrow c'] \models sv_1$.

We can now state the main property: the set of satisfiable symbolic models (with respect to a formula ϕ) is recognized by the simple one-$\mathbb{Z}$-counter automaton $\mathcal{A}_{\mathrm{sat}}$.

Let $\langle \sigma, \rho \rangle$ be a satisfiable symbolic model and for all $i \in \mathbb{N}$, $\rho(i) = \langle (\mathrm{E}_x)_i, (\mathrm{E}_m)_i, (\mathrm{E}'_x)_i, (\mathrm{E}'_m)_i, (\mathrm{E}_s)_i \rangle$. So there is a $\mathrm{CLTL}^1_1(\mathrm{DL})$ model $\sigma' : \mathbb{N} \to \mathbb{Z}$ such that $\sigma' \models \rho$. We can show that $\langle \sigma, \rho \rangle \in \mathrm{L}(\mathcal{A}_{\mathrm{sat}})$ since there is an accepting run of the form

$r_0 \xrightarrow{\varepsilon} \langle q_{in}^{(\mathrm{E}_m)_0, \rho(0)}, \sigma'(0) \rangle \xrightarrow{\varepsilon}{}^* \langle q_{in}^{(\mathrm{E}_x)_0, \rho(0)}, \sigma'(0) \rangle \xrightarrow{\varepsilon}{}^* \langle q_{in}^{\langle (\mathrm{E}'_m)_0, (\mathrm{E}_s)_0 \rangle, \rho(0)}, \sigma'(0) \rangle \xrightarrow{\varepsilon}{}^* \langle q_{out}^{\langle (\mathrm{E}'_m)_0, (\mathrm{E}_s)_0 \rangle, \rho(0)}, \sigma'(1) \rangle \xrightarrow{\rho(0)} \langle q_{in}^{(\mathrm{E}_x)_1, \rho(1)}, \sigma'(1) \rangle \xrightarrow{\varepsilon}{}^* \langle q_{in}^{\langle (\mathrm{E}'_m)_1, (\mathrm{E}_s)_1 \rangle, \rho(1)}, \sigma'(1) \rangle \xrightarrow{\varepsilon}{}^* \langle q_{out}^{\langle (\mathrm{E}'_m)_1, (\mathrm{E}_s)_1 \rangle, \rho(1)}, \sigma'(2) \rangle \xrightarrow{\rho(1)} \langle q_{in}^{(\mathrm{E}_x)_2, \rho(2)}, \sigma'(2) \rangle \dots$

Now suppose that $\langle \sigma, \rho \rangle \in \mathrm{L}(\mathcal{A}_{\mathrm{sat}})$. By construction of the network of components in $\mathcal{A}_{\mathrm{sat}}$, there is an accepting run necessarily of the form

$r_0 \xrightarrow{\varepsilon} \langle q_{in}^{(\mathrm{E}_m)_0, sv_0}, c_0 \rangle \xrightarrow{\varepsilon}{}^* \langle q_{in}^{(\mathrm{E}_x)_0, sv_0}, c_0 \rangle \xrightarrow{\varepsilon}{}^* \langle q_{in}^{\langle (\mathrm{E}'_m)_0, (\mathrm{E}_s)_0 \rangle, sv_0}, c_0 \rangle \xrightarrow{\varepsilon}{}^* \langle q_{out}^{\langle (\mathrm{E}'_m)_0, (\mathrm{E}_s)_0 \rangle, sv_0}, c_1 \rangle \xrightarrow{\rho(0)} \langle q_{in}^{(\mathrm{E}_x)_1, sv_1}, c_1 \rangle \xrightarrow{\varepsilon}{}^* \langle q_{in}^{\langle (\mathrm{E}'_m)_1, (\mathrm{E}_s)_1 \rangle, sv_1}, c_1 \rangle \xrightarrow{\varepsilon}{}^* \langle q_{out}^{\langle (\mathrm{E}'_m)_1, (\mathrm{E}_s)_1 \rangle, sv_1}, c_2 \rangle \xrightarrow{\rho(1)} \langle q_{in}^{(\mathrm{E}_x)_2, sv_1}, c_2 \rangle \dots$

By construction of each component and by equivalence between (I) and (II) above, $[x \leftarrow c_0] \models (\mathrm{E}_m)_0$ and for every $i \in \mathbb{N}$, $[x \leftarrow c_i, \mathsf{X}x \leftarrow c_{i+1}] \models (\mathrm{E}_x)_i \wedge (\mathrm{E}_m)_i \wedge (\mathrm{E}'_x)_i \wedge (\mathrm{E}'_m)_i \wedge (\mathrm{E}_s)_i$. So the model $\sigma' : \mathbb{N} \to \mathbb{Z}$ s.t. $\sigma'(i) = c_i$ satisfies ρ, i.e. $\sigma' \models \rho$. This means precisely that $\langle \sigma, \rho \rangle$ is a satisfiable symbolic model. □

In order to evaluate the complexity of the nonemptiness test for $\mathcal{A}_\phi$, we also need the following result.

Theorem 2 *The nonemptiness problem for simple one-$\mathbb{Z}$-counter automata with alphabet is* NLOGSPACE*-complete.*

The tedious proof of Theorem 2 (see [14, Sect. 6]) is based on the two following results. First, checking whether $\mathrm{L}(\mathcal{A})$ is non-empty for simple one-$\mathbb{Z}$-counter automata $\mathcal{A}$ with alphabet can be reduced in logarithmic space to the existence of an accepting run in simple one-$\mathbb{N}$-counter automata with no test $x \neq 0$ and no alphabet (easy). Second, checking the nonemptiness of automata from the latter class amounts to check the existence of paths of polynomial lengths satisfying specific properties. This second part requires careful and lengthy developments. We are now ready to state the main complexity result and its main corollary.

Theorem 3 *Satisfiability for* $\mathrm{CLTL}^1_1(\mathrm{DL}^+, \mathrm{PROP})$ *is* PSPACE*-complete.*

The presence of propositional variables in $\mathrm{CLTL}(\mathrm{DL}^+, \mathrm{PROP})$ allows to reduce the model-checking problem for $\mathrm{CLTL}(\mathrm{DL}^+)$ to the satisfiability problem for $\mathrm{CLTL}(\mathrm{DL}^+, \mathrm{PROP})$ (following [29]). By inspection of the proof (see [14, Sect. 4.2]) we obtain a logspace reduction from the model-checking problem for $\mathrm{CLTL}^1_1(\mathrm{DL}^+)$ to the satisfiability problem for $\mathrm{CLTL}^1_1(\mathrm{DL}^+, \mathrm{PROP})$.

Corollary 2 *Satisfiability for* $\mathrm{CLTL}^1_1(\mathrm{DL}^+)$ *and model-checking* $\mathrm{CLTL}^1_1(\mathrm{DL}^+)$ *formulae over 1-variable* DL^+*-automata are* PSPACE*-complete.*

PSPACE-hardness follows from PSPACE-hardness of satisfiability and model-checking for LTL restricted to one variable [15]. As additional corollaries, we deduce that the one-variable fragment of the counter logic

$\mathcal{L}_p$ [10] has a PSPACE-complete satisfiability problem and model-checking one-clock discrete timed automata with $\mathrm{CLTL}^1_1(\mathrm{DL}^+)$ can be done in PSPACE which contrasts with the undecidability results from [11, Section 6]. Corollary 2 can be extended by allowing propositional variables in the automata and formulae. More importantly, a quite remarkable separation feature of our technique is that we can adapt it to any extension LTL^+ of LTL for which formulae can be translated into Büchi automata in polynomial space. This includes extensions with past-time operators, with automata-based operators [34], or with fixpoint operator, see e.g. [30].

Theorem 4 $(\mathrm{LTL}^+)^1_1(\mathrm{DL}^+)$ *model-checking and satisfiability are also in* PSPACE.

It suffices to adapt the definition of $\mathcal{A}_{\mathrm{symb}}$ from plain LTL to LTL^+, the automaton $\mathcal{A}_{\mathrm{sat}}$ from Lemma 3 being unchanged. As a corollary, model-checking linear μ-calculus with difference logic DL over one-variable DL-automata is PSPACE-complete, refining a result from [8].

We conclude this section by a more prospective remark. Bounded model-checking [4] consists in searching for a counterexample in executions whose length is bounded by some integer m (encoded in binary). By adding to a finite-state system a counter that increments after each transition, one can concisely encode in our formalism the problem of finding a witness execution of length m. Of course, one needs to relativize the formulae: for instance $p\mathsf{U}q$ would become $(x < m \Rightarrow p)\mathsf{U}(x < m \wedge q)$.

4 Model-checking one-$\mathbb{Z}$-counter automata

A natural question is whether Corollary 2 is optimal w.r.t. the Presburger fragment we have considered. Lemma 4 below states that we do not preserve decidability when extending the constraint language to QFP.

Lemma 4 *Satisfiability for* $\mathrm{CLTL}^1_1(\mathrm{QFP})$ *and model-checking* $\mathrm{CLTL}^1_1(\mathrm{QFP})$ *formulae over 1-variable* QFP*-automata are* Σ^1_1*-complete.*

Constraints of the form $ax + by = 0$ where $a, b \in \mathbb{Z}$ allow to encode a configuration $\langle q_i, c_1, c_2\rangle$ of a Minsky machine (q_i is the ith control state) by the value $2^{c_1}3^{c_2}5^i$ and zero tests can be done using modulo relations. This follows directly from [26, Sect. 14.2] about one counter machines with multiplication and division by constants. In this section, the strategy to regain decidability consists in restricting the class of models to one-$\mathbb{Z}$-counter automata. Model-checking becomes decidable (even in PSPACE) for LTL with *full quantifier-free Presburger constraints* restricted to *one variable* but with *no restriction on the* X*-length*.

Let $\mathcal{A} = \langle Q_{\mathcal{A}}, \delta_{\mathcal{A}}, I_{\mathcal{A}}, F_{\mathcal{A}}\rangle$ be a one-$\mathbb{Z}$-counter automaton (not necessarily simple) whose set of updates is of the form $\mathsf{X}x = x + u$ with $u \in \{u_{\min}, u_{\min}+1, \ldots, u_{\max}\}$. Without any loss of generality, we can assume that $u_{\min} = -u_{\max}$. Given a $\mathrm{CLTL}^\omega_1(\mathrm{QFP})$ formula ϕ such that $|\phi|_{\mathsf{X}} = l$, we consider the following syntactic resources:

- K is the lcm of the integers k such that $\equiv_k$ occurs in ϕ,
- CONS is the set of constants d such that $\sum a_i\mathsf{X}^i x \sim d$ occurs in ϕ,
- COEF is the set of constants a_i such that $\sum a_i\mathsf{X}^i x \sim d$ occurs in ϕ.

Without any loss of generality, we can assume that $a_{\min} = -a_{\max}$ where $a_{\min}$ and $a_{\max}$ are respectively the minimal and the maximal element of COEF. For technical reasons (see details in [14, Sect. 5]), we pose $\mathrm{CONS}(\mathcal{A}, \phi) = \{d_{\min}, \ldots, d_{\max}\}$ such that $d_{\max} = -d_{\min} = \frac{l(l+1)}{2}a_{\max}u_{\max}$. We define a symbolic valuation wrt $\mathcal{A}$ and ϕ as an element of the set $SV(\mathcal{A}, \phi) = \mathrm{C}_x \times \mathrm{Mod}_x \times \mathrm{C}^1_{\mathrm{step}} \times \cdots \times \mathrm{C}^l_{\mathrm{step}}$ such that

- C_x is the set composed of the constraints $x < d_{\min}$, $d_{\max} < x$, and $x = d$ for $d \in \mathrm{CONS}(\mathcal{A}, \phi)$.
- Mod_x is the set composed of the constraints $x \equiv_K c$ for $c \in \{0, \ldots, K-1\}$.
- $\mathrm{C}^i_{\mathrm{step}}$ is composed of constraints $\mathsf{X}^i x = \mathsf{X}^{i-1}x + u$ for $u \in \{u_{\min}, \ldots, u_{\max}\}$.

A result similar to Lemma 1 can be established.

We define a symbolic satisfaction relation as in Sect. 3: $sv \models_{\mathrm{symb}} \mathrm{E}$ iff for every valuation $v : \{x, \mathsf{X}x, \ldots, \mathsf{X}^l x\} \to \mathbb{Z}$ such that $sv(v) = sv$ we have $v \models \mathrm{E}$. We can naturally extend this relation to symbolic valuation sequences.

Lemma 5 *Let* ϕ *be a* $\mathrm{CLTL}^\omega_1(\mathrm{QFP})$ *formula and* $\mathcal{A}$ *be a one-*$\mathbb{Z}$*-counter automaton.* $\mathcal{A} \models \phi$ *iff there are a symbolic model* $\rho \in SV(\mathcal{A}, \phi)$ *such that* $\rho \models_{\mathrm{symb}} \phi$ *and an accepting run* $\langle q_0, c_0\rangle, \langle q_1, c_1\rangle, \langle q_2, c_3\rangle, \ldots$ *of* $\mathcal{A}$ *such that* $c_0, c_1, c_2 \ldots \models \rho$.

We build the automaton $\mathcal{A}_\phi$ as the intersection $\mathcal{A}_{\mathrm{symb}} \cap \mathcal{A}_{\mathrm{sat}}$ such that $\mathcal{A}_{\mathrm{symb}}$ recognizes the set of symbolic models satisfying ϕ and $\mathcal{A}_{\mathrm{sat}}$ recognizes the set of symbolic models generated from accepting runs. The definition of $\mathcal{A}_{\mathrm{symb}}$ and the synchronization between $\mathcal{A}_{\mathrm{symb}}$ and $\mathcal{A}_{\mathrm{sat}}$ are similar to Sect. 3 considering the alphabet $\Sigma = \mathrm{SV}(\mathcal{A}, \phi)$ (with ε-transitions) and the corresponding relation $\models_{\mathrm{symb}}$. Lemma 6 below is a pivot result for proving Theorem 5 and its proof is a variant of the proof of Lemma 3.

Lemma 6 *Given a formula* ϕ *and a one-*$\mathbb{Z}$*-counter automaton* $\mathcal{A}$*, one can build a simple one-*$\mathbb{Z}$*-counter automaton*

$\mathcal{A}_{\rm sat}$ *with alphabet* $\Sigma = 2^{\rm PROP} \times {\rm SV}(\mathcal{A},\phi)$ *such that* ${\rm L}(\mathcal{A}_{\rm sat})$ *is the set of satisfiable symbolic models w.r.t* ϕ *and* $\mathcal{A}$*.*
Moreover, $\mathcal{A}_\phi$ *can be effectively built from* ϕ *and* $\mathcal{A}$ *in polynomial space thanks to the way* $\mathcal{A}_{\rm sat}$ *is defined.*

Proof. Let $\mathcal{A} = \langle Q_\mathcal{A}, I_\mathcal{A}, F_\mathcal{A}, \delta_\mathcal{A}\rangle$ be a one $\mathbb{Z}$-counter automaton and ϕ be a ${\rm CLTL}_1^\omega({\rm QFP})$ formula such that $|\phi|_{\mathsf{X}} = l$. The automaton $\mathcal{A}_{\rm sat}$ is defined over the alphabet $\Sigma = SV(\mathcal{A},\phi)$. As in the proof of Lemma 3, the construction is modular. We denote $Q_\mathcal{A} \times Q$ the set of states of $\mathcal{A}_{\rm sat}$ where Q is a set of auxiliary states used in the construction of components similar to components in the proof of Lemma 3. For every $sv = \langle {\rm E}_x, {\rm E}_m, {\rm E}_s^1, \ldots, {\rm E}_s^l\rangle$ and $q_a \in Q_\mathcal{A}$, we define components:

- $\mathcal{A}^{q_a}_{{\rm E}_m,sv}$ such that for every $c \in \mathbb{Z}$, $\langle q_a, q_{in}^{{\rm E}_m,sv}, 0\rangle \rightarrow^* \langle q_a, q_{out}^{{\rm E}_m,sv}, c\rangle$ iff $[x \leftarrow c] \models {\rm E}_m$.
- $\mathcal{A}^{q_a}_{{\rm E}_x,sv}$ such that for every $c \in \mathbb{Z}$, $\langle q_a, q_{in}^{{\rm E}_x,sv}, c\rangle \rightarrow^* \langle q_a, q_{out}^{{\rm E}_x,sv}, c'\rangle$ iff $c = c'$ and $[x \leftarrow c] \models {\rm E}_x$.
- We also need to define another kind of components that update the counter. For every $d \in {\rm CONS}(\mathcal{A},\phi)$, $\mathcal{A}^{q_a}_{{\rm E}_s^1,sv}$ is such that for every $c \in \mathbb{Z}$, $\langle q_a, q_{in}^{{\rm E}_s^1,sv}, c\rangle \rightarrow^* \langle q_a, q_{out}^{{\rm E}_s^1,sv}, c'\rangle$ iff $[x \leftarrow c, \mathsf{X}x \leftarrow c'] \models {\rm E}_s^1$.

The different components are connected as follows:

- The set of initial states is composed of the states of the form $\langle q_0, q_{in}^{{\rm E}_m,sv}\rangle$ for every $sv = \langle {\rm E}_x, {\rm E}_m, {\rm E}_s^1, \ldots, {\rm E}_s^l\rangle \in {\rm SV}(\phi,\mathcal{A})$ and $q_0 \in I_\mathcal{A}$.
- For every $sv = \langle {\rm E}_x, {\rm E}_m, {\rm E}_s^1, .., {\rm E}_s^l\rangle \in {\rm SV}(\phi,\mathcal{A})$ and $q_0 \in I_\mathcal{A}$, $\langle q_0, q_{out}^{{\rm E}_m,sv}\rangle \xrightarrow{\varepsilon,\top,0} \langle q_0, q_{in}^{{\rm E}_x,sv}\rangle \in \delta$.
- For all $sv = \langle {\rm E}_x, {\rm E}_m, {\rm E}_s^1, \ldots, {\rm E}_s^l\rangle \in {\rm SV}(\phi,\mathcal{A})$, $\langle q_a, q_{out}^{{\rm E}_x,sv}\rangle \xrightarrow{\varepsilon,\top,0} \langle q_a, q_{in}^{{\rm E}_s^1,sv}\rangle \in \delta$.
- For all $sv_1 = \langle ({\rm E}_x)_1, ({\rm E}_m)_1, ({\rm E}_s^1)_1, .., ({\rm E}_s^l)_1\rangle \in {\rm SV}(\phi,\mathcal{A})$, $sv_2 = \langle ({\rm E}_x)_2, ({\rm E}_m)_2, ({\rm E}_s^1)_2, .., ({\rm E}_s^l)_2\rangle \in {\rm SV}(\phi,\mathcal{A})$ and $q_a, q'_a \in Q_\mathcal{A}$ such that
  - $({\rm E}_s^1)_1$ equals $\mathsf{X}x = x + d$ and $q_a \xrightarrow{\sim,d} q'_a \in \delta_\mathcal{A}$ for some $\sim \in \{\top, <, >, =, \neq\}$,
  - $({\rm E}_s^1)_1 \wedge ({\rm E}_m^1)_1 \Rightarrow ({\rm E}_m^1)_2$ is valid in PA (checkable in polynomial time),
  - for every $1 \leq i \leq l-1$, $({\rm E}_s^i)_2 = ({\rm E}_s^{i+1})_1[\mathsf{X}^{i+1}x \leftarrow \mathsf{X}^i x, \mathsf{X}^i x \leftarrow \mathsf{X}^{i-1}x]$,

  we have $\langle q_a, q_{out}^{{\rm E}_s^1,sv_1}\rangle \xrightarrow{sv_1,\sim,0} \langle q'_a, q_{in}^{({\rm E}_x)_2,sv_2}\rangle \in \delta$.
- The set of final states is $\{\langle q_f, q_{out}^{{\rm E}_x,sv}\rangle \mid q_f \in F_\mathcal{A}\}$.

By construction, for all paths $\cdots \langle q_i, q_{in}^{({\rm E}_x)_i,sv_i}, c_i\rangle \rightarrow \cdots \xrightarrow{sv_i} \langle q_{i+1}, q_{in}^{({\rm E}_x)_{i+1},sv_{i+1}}, c_{i+1}\rangle \cdots \xrightarrow{sv_{i+1}} \cdots \xrightarrow{sv_{i+l-1}} \langle q_{i+l}, q_{in}^{({\rm E}_x)_{i+l},sv_{i+l}}, c_{i+l}\rangle \cdots$ in $\mathcal{A}_{\rm sat}$, we have $c_i, \ldots, c_{i+l} \models sv_i$. □

Now we can conclude about the complexity of the model-checking problem.

Theorem 5 *Model-checking* ${\rm CLTL}_1^\omega({\rm QFP})$ *formulae over one-*$\mathbb{Z}$*-counter automata is* PSPACE*-complete.*

PSPACE-hardness is a consequence of [15], see also a direct proof in [14, Sect. 5]. As for Theorem 4, model-checking linear μ-calculus with QFP constraints over one-$\mathbb{Z}$-counter automata is in PSPACE, refining [8]. By contrast, satisfiability for ${\rm CLTL}_1^1({\rm QFP})$ is undecidable.

5 Conclusion

Figure 4 summarizes the complexity of satisfiability, model-checking over DL-automata and model-checking over k-$\mathbb{Z}$-counter automata for most LTL-like specification languages considered herein.

Apart from the completion of our classification, the more positive results concern one-counter automata/nets, see applications in [9, 32, 21]. The PSPACE upper bound for model-checking one-$\mathbb{Z}$-counter automata over ${\rm CLTL}_1^\omega({\rm QFP})$ or even over its linear μ-calculus extension refines results from [18, 8, 33, 28] that concern more general systems and languages.

References

[1] R. Alur and T. Henzinger. A really temporal logic. *JACM*, 41(1):181–204, 1994.

[2] P. Balbiani and J. Condotta. Computational complexity of propositional linear temporal logics based on qualitative spatial or temporal reasoning. In *FROCOS'02*, volume 2309 of *LNAI*, pages 162–173. Springer, 2002.

[3] S. Bardin, A. Finkel, E. Lozes, and A. Sangnier. From pointer systems to counter systems using shape analysis. In *AVIS'06*, 2006.

[4] A. Biere, A. Cimatti, E. Clarke, O. Strichman, and Y. Zhu. Bounded model checking. In vol. 58 of Advances in Computers. Academic Press, 2003. To appear.

[5] B. Boigelot. *Symbolic methods for exploring infinite state spaces*. PhD thesis, Université de Liège, 1998.

[6] A. Bouajjani, M. Bozga, P. Habermehl, R. Iosif, P. Moro, and T. Vojnar. Programs with lists are counter automata. In *CAV'06*, volume 4144 of *LNCS*, pages 517–531. Springer, 2006.

[7] A. Bouajjani, R. Echahed, and P. Habermehl. On the verification problem of nonregular properties for nonregular processes. In *LICS'95*, pages 123–133, 1995.

	MC (DL)	SAT	MC (CA)
$\mathrm{CLTL}_3^1(\mathrm{DL})$	undec. [10]	undec. [10]	undec. [10]
$\mathrm{CLTL}_2^\omega(\mathrm{DL})$	undec. [26]	undec. [12]	undec. [26]
$\mathrm{CLTL}_1^2(\mathrm{DL})$	undec., Cor. 1	undec., Theo. 1	PSPACE-c, Theo. 5 + [15]
$\mathrm{CLTL}_2^1(\mathrm{DL})$	undec. [26]	undec., Theo. 1	undec. [26]
$\mathrm{CLTL}_1^1(\mathrm{DL\ or\ DL}^+)$	PSPACE-c., Cor. 2	PSPACE-c., Cor. 2	PSPACE-c, Theo. 5 + [15]
$\mathrm{CLTL}_1^1(\mathrm{QFP})$	undec., Lem. 4	undec., Lem. 4	PSPACE-c, Theo. 5 + [15]
$\mathrm{CLTL}_1^\omega(\mathrm{QFP})$	undec., Lem. 4	undec., Lem. 4	PSPACE-c., Theo. 5 + [15]

Figure 4. Summary

[8] A. Bouajjani, J. Esparza, and O. Maler. Reachability analysis of pushdown automata: Application to model checking. In *CONCUR'97*, volume 1243 of *LNCS*, pages 135–150. Springer, 1997.

[9] C. Chitic and D. Rosu. On validation of XML streams using finite state machines. In *WebDB, Paris*, pages 85–90, 2004.

[10] H. Comon and V. Cortier. Flatness is not a weakness. In *CSL'00*, volume 1862 of *LNCS*, pages 262–276. Springer, 2000.

[11] Z. Dang, P. S. Pietro, and R. Kemmerer. Presburger liveness verification of discrete timed automata. *TCS*, 299:413–438, 2003.

[12] S. Demri and D. D'Souza. An automata-theoretic approach to constraint LTL. *I & C*, 205(3):380–415, 2007.

[13] S. Demri and R. Gascon. Verification of qualitative $\mathbb{Z}$-constraints. In *CONCUR'05*, volume 3653 of *LNCS*, pages 518–532. Springer, 2005.

[14] S. Demri and R. Gascon. The effects of bounding syntactic resources on Presburger LTL. Technical Report LSV-06-5, LSV, 2006.

[15] S. Demri and P. Schnoebelen. The complexity of propositional linear temporal logics in simple cases. *I & C*, 174(1):84–103, 2002.

[16] A. Finkel and J. Leroux. How to compose Presburger accelerations: Applications to broadcast protocols. In *FST&TCS'02*, volume 2256 of *LNCS*, pages 145–156. Springer, 2002.

[17] A. Finkel and G. Sutre. Decidability of reachability problems for classes of two counters automata. In *STACS'00*, volume 2256 of *LNCS*, pages 346–357. Springer, 2000.

[18] A. Finkel, B. Willems, and P. Wolper. A direct symbolic approach to model checking pushdown systems (extended abstract). In *INFINITY'97*, volume 9 of *ENTCS*, pages 27–39. Elsevier Science, 1997.

[19] O. Ibarra, J. Su, Z. Dang, T. Bultan, and A. Kemmerer. Counter machines: Decidable properties and applications to verification problems. In *MFCS'00*, volume 1893 of *LNCS*, pages 426–435. Springer, 2000.

[20] O. Kupferman, M. Y. Vardi, and P. Wolper. An automata-theoretic approach to branching-time model checking. *JACM*, 47(2):312–360, 2000.

[21] P. Lafourcade, D. Lugiez, and R. Treinen. Intruder deduction for AC-like equational theories with homomorphisms. In *RTA'05*, volume 3467 of *LNCS*, pages 308–322. Springer, 2005.

[22] F. Laroussinie, N. Markey, and P. Schnoebelen. Model checking timed automata with one or two clocks. In *CONCUR'04*, volume 3170 of *LNCS*, pages 387–401. Springer, 2004.

[23] S. Lasota and I. Walukiewicz. Alternating timed automata. In *FOSSACS'05*, volume 3441 of *LNCS*, pages 250–265. Springer, 2005.

[24] J. Leroux and G. Sutre. Flat counter systems are everywhere! In *ATVA'05*, volume 3707 of *LNCS*, pages 489–503. Springer, 2005.

[25] C. Lutz. NEXPTIME-complete description logics with concrete domains. *ACM Transactions on Computational Logic*, 5(4):669–705, 2004.

[26] M. Minsky. *Computation, Finite and Infinite Machines*. Prentice Hall, 1967.

[27] J. Ouaknine and J. Worrell. On the decidability of metric temporal logic. In *LICS'05*, pages 188–197, 2005.

[28] O. Serre. Parity games played on transition graphs of one-counter processes. In *FOSSACS'06*, volume 3921 of *LNCS*, pages 337–351. Springer, 2006.

[29] A. Sistla and E. Clarke. The complexity of propositional linear temporal logic. *JACM*, 32(3):733–749, 1985.

[30] M. Vardi. A temporal fixpoint calculus. In *POPL'88*, pages 250–259. ACM, 1988.

[31] M. Vardi and P. Wolper. Reasoning about infinite computations. *I & C*, 115:1–37, 1994.

[32] M. Wakatsuki, K. Teraguchi, and E. Tomita. Polynomial time identification of strict deterministic restricted one-counter automata in some class from positive data. In *ICGI'04*, volume 3264 of *LNAI*, pages 260–272. Springer, 2004.

[33] I. Walukiewicz. Pushdown processes: games and model-checking. *I & C*, 164(2):234–263, 2001.

[34] P. Wolper. Temporal logic can be more expressive. *I & C*, 56:72–99, 1983.