LTL over Integer Periodicity Constraints*

Stéphane Demri

LSV/CNRS UMR 8643 & INRIA Futurs projet SECSI & ENS Cachan
61, av. Pdt. Wilson, 94235 Cachan Cedex, France
email: demri@lsv.ens-cachan.fr

February 24th, 2004

Abstract. Periodicity constraints are used in many logical formalisms, in fragments of Presburger LTL, in calendar logics, and in logics for access control, to quote a few examples. In the paper, we introduce the logic $\mathrm{PLTL}^{\mathrm{mod}}$, an extension of Linear-Time Temporal Logic LTL with past-time operators whose atomic formulae are defined from a first-order constraint language dealing with periodicity. Although the underlying constraint language is a fragment of Presburger arithmetic shown to admit a PSPACE-complete satisfiability problem, we establish that $\mathrm{PLTL}^{\mathrm{mod}}$ model-checking and satisfiability problems remain in PSPACE as plain LTL (full Presburger LTL is known to be highly undecidable). This is particularly interesting for dealing with periodicity constraints since the language of $\mathrm{PLTL}^{\mathrm{mod}}$ has a language more concise than existing languages and the temporalization of our first-order language of periodicity constraints has the same worst case complexity as the underlying constraint language. Finally, we show that adding quantification in the logical language provides to $\mathrm{PLTL}^{\mathrm{mod}}$, EXPSPACE-hard problems. As another application, we establish that the equivalence problem for extended single-string automata, known to express the equality of time granularities, is PSPACE-complete by designing a reduction from QBF and by using our results for $\mathrm{PLTL}^{\mathrm{mod}}$. The paper concludes by presenting a bunch of open problems related to decidability and complexity issues for other fragments of Presburger LTL.

Key-words: Presburger LTL, periodicity constraints, computational complexity, Büchi automaton, QBF.

1 Introduction

Presburger Constraints. Presburger constraints (see e.g., [Pre29]) are present in many logical formalisms including extensions of Linear-Time Logic LTL. We quote below some examples:

– Timed Propositional Temporal Logic(s) [AH94];

* Extended version of [Dem04].

– CTL* dedicated to integral relational automata [Čer94];
– Constrained LTL named CLTL defined with LTL models but with Presburger occurrences constraints [BEH95];
– Flat fragment of Presburger LTL [CC00] (see also [CJ98]).

Other formalisms more dedicated to formal verification can be found in

– model-checking of discrete timed automata [DPK03];
– verification of infinite-state systems with linear arithmetic constraints, see e.g. [BH99,WB00,Wol01,FS00,FL02,BB03].

In the paper, we are interested in models of Presburger LTL that are ω-sequences of valuations for a given set VAR of integer variables taking their values in $\mathbb{Z}$ and the atomic formulae are Presburger arithmetic constraints with free variables in VAR (the models in [BEH95] are quite different since they are just plain LTL models). For instance, $\phi = \Box(\mathsf{X}x = x)$ states that the value of the variable x is constant over the time line where $\mathsf{X}x$ denotes the value of x at the next state. A model of ϕ is simply an ω-sequence in $(\mathbb{Z})^\omega$. The counterpart of the high expressive power of Presburger LTL rests on its undecidability, shown by a standard encoding of the halting problem for two-counter machines. However, to regain decidability one can either restrict the underlying constraint language, see e.g. [AH94, Sect. 3] and [DD02], or restrict the logical language, see e.g. a decidable flat fragment of Presburger LTL in [CC00]. Herein, we shall consider versions of LTL with Presburger constraints with the *full logical language* (mainly LTL with past-time operators sometimes augmented with first-order quantifiers) but with *strict fragments of Presburger arithmetic*. As a consequence, all the constraint languages we will consider are closed under Boolean operations.

Our motivations. Integer periodicity constraints, a special class of Presburger constraints, have found applications in many logical formalisms such as

– DATALOG with integer periodicity constraints [TC98];
– logical formalisms dealing with calendars, see e.g. [Ohl94,Wij00,CFP02];
– temporal reasoning in database access control [BBFS96,BBFS98];
– periodic time in generalized databases, see e.g. [Wol01,NS92].

Moreover, abstracting programs with integer variables by constraint automata with periodicity constraints can be viewed as a way to handle the analysis of such programs. Although we will not elaborate on it in this paper, we believe it is a promising continuation of the current paper, in the line of [MOS04] for instance.
In view of the ubiquity of integer periodic constraints, the main motivation of the current work is to design a variant of LTL over a language for integer periodicity constraints that satisfies the following properties.

1. The logical language contains at least LTL (no flatness restriction).

2. The constrained language is expressive enough to capture most integer periodicity constraints used in calendar logics and in database access control. For instance, in [CFP02], the authors advocate the need to design an extension of LTL that expresses quantitative temporal requirements, such as periodicity constraints. We provide in the paper such an extension.
3. Model-checking and satisfiability remain in PSPACE and possibly to be able to adapt the technique with Büchi automata [VW94] to this new extension of LTL.

Last but not least, as a long-term project, we wish to understand what are the decidable fragments of Presburger LTL by restricting the constraint language but with the full logical language.

Our contribution. We introduce a decidable fragment of Presburger LTL that satisfies the above-mentioned requirements. Let us be a bit more precise.

1. We introduce a first-order theory of integer periodicity constraints IPC^{++} and we show its PSPACE-completeness. This is a fragment of Presburger arithmetic that extends the one from [TC98].
2. We show the PSPACE-completeness of PLTL (LTL with past-time operators) over IPC^{++} using Büchi automata (logic denoted by $\mathrm{PLTL}^{\mathrm{mod}}$ in the paper) in the line of [VW94].
3. We demonstrate why adding the existential operator $\exists$ at the logical level ($\exists$ is already present at the constraint level) leads to an exponential blow-up of the complexity. We show that $\mathrm{PLTL}(\mathrm{IPC}^{+})$ augmented with $\exists$ is in EXPSPACE and $\mathrm{PLTL}^{\mathrm{mod}}$ augmented with $\exists$ is EXPSPACE-hard. IPC^{+} is a fragment of IPC^{++} without constraints of the form $x = y$.
4. As an application, we show the PSPACE-completeness of the equivalence problem for the extended single-string automata [LM01, Sect. 5]. Extended single-string automata are Büchi automata that recognize exactly one ω-word and guards involving periodicity constraints are present on the transitions. This formalism has been introduced as a concise means to define time granularities and the equivalence problem for such automata is central to check the equality of time granularities, see also [Wij00]. Roughly speaking, a time granularity is a mapping from integer numbers to subsets of a time domain.

Related work. Apart from the above-mentioned works dealing with (fragments of) Presburger LTL, we quote below some related works involving periodicity constraints, fragments of Presburger arithmetic, versions of LTL over concrete domains, and constraint automata.

1. In [TC98] a version of DATALOG with integer periodicity constraints is studied. Our constraint language extends the one from [TC98] but is used in a different way since it serves as a basis for the atomic formulae of our studied fragment of Presburger LTL. Similar periodicity constraints can be found in [BBFS96,BBFS98].

2. Complexity issues for versions of LTL over concrete domains have been studied in [BC02,DD03,GKWZ03,GKK$^+$03] (see also [Lut01]). Unlike most of these works, we use a computationally expensive constraint language (satisfiability is PSPACE-complete) while preserving the PSPACE upper bound of the corresponding fragment of Presburger LTL.
3. Various fragments of Presburger arithmetic have been introduced in the literature, very often for quite different purposes. By way of example, let us mention the PSPACE-complete fragment of Presburger arithmetic introduced in [Kou94] with quantifier elimination. OCP "One counter Properties" [JKMS02] is also a fragment of Presburger arithmetic that is DP-hard, and in Π_2^p (level 2 of polynomial hierarchy). A peculiarity of OCP is that it lacks the closure under negation.
4. Constraint automata similar to the one we shall consider in this paper are introduced in [LM01] (extended single-string automata). The underlying constraint language L_I from [LM01] is actually a fragment of the language IPC$^+$ (see Sect. 6 for more details). Decidability of the equivalence of two such constraint automata is shown decidable in [LM01] by using exponential space. We shall explain why the extended single-string automata can be viewed as a subclass of LTL(IPC$^+$)-automata (defined in Sect. 2.4).

Plan of the paper. The rest of the paper is structured as follows. In Sect. 2 we introduce fragments of Presburger arithmetic involving periodicity constraints and the corresponding versions of LTL with past defined upon them. Sect. 3 is dedicated to the most expressive constraint language considered in the paper, namely IPC^{++}. We show that IPC^{++} admits quantifier elimination in polynomial space, satisfiability is PSPACE-complete and we provide a symbolic representation for solutions of IPC^{++} constraints. In Sect. 4 we show that PLTL$^{\text{mod}}$ satisfiability and model-checking are PSPACE-complete problems by taking advantage of results from Sect. 3 to abstract PLTL$^{\text{mod}}$ models of a given PLTL$^{\text{mod}}$ formula. These problems are reduced to the emptiness problem for Büchi automata. Sect. 5 analyses the complexity of the logic PLTL(IPC$^+$), a fragment of PLTL$^{\text{mod}}$, augmented with the quantifier $\exists$ at the logical level. We show that PLTL(IPC$^+$) augmented with $\exists$ has a satisfiability problem in EXPSPACE. Sect. 6 presents the proof of the PSPACE-completeness of the equivalence problem for extended single-string automata. In Sect. 7, we provide concluding remarks and underline the open problems we consider worth being tackled.

2 PLTL over periodicity constraints

2.1 Constraint languages

Let VAR $= \{x_0, x_1, \ldots\}$ be a countably infinite set of variables. We define below languages of the first-order theory of integer periodicity constraints. The constraint language IPC is defined by the grammar below:

$$p ::= x \equiv_k y + c \;\mid\; x \equiv_k c \;\mid\; p \wedge p \;\mid\; \neg p,$$

where $k, c \in \mathbb{N}$. A simple periodicity constraint is a conjunction of constraints of the form either $x \equiv_k y + c$ or $x \equiv_k c$. Given $X \subseteq \{\exists, [], <, =\}$, we define an extension of IPC, namely IPC^X, by adding clauses to the definition of IPC:

- if $\exists \in X$, then the clause $\exists\ x\ p$ is added (existential quantification);
- if $[] \in X$, then the clause $x \equiv_k y + [c_1, c_2]$ with $c_1, c_2 \in \mathbb{N}$ is added;
- if $= \in X$, then the clause $x = y$ with $x, y \in \mathrm{VAR}$ is added;
- if $< \in X$, then the clauses $x < c \ \mid\ x > c \ \mid\ x = c$ with $x \in \mathrm{VAR}$ and $c \in \mathbb{Z}$ are added.

In the sequel, IPC^+ denotes $\mathrm{IPC}^{\{\exists,[],<\}}$ whereas IPC^{++} denotes $\mathrm{IPC}^{\{\exists,[],<,=\}}$, which is actually the richer constraint language considered in the paper. IPC^{++} is the extension of the language of the first-order theory of integer periodicity constraints introduced in [TC98] but with the inclusion of negation as considered in [BBFS96]. Unlike [TC98], we shall not use periodicity graphs as symbolic representation of sets of tuples definable by constraints in IPC (see also the complementation of periodicity graphs in [BBFS96]). Instead, we shall represent periodicity constraints by sets of triples of natural numbers. The cardinality of such sets will be exponential in the size of the corresponding constraints (see details in Sect. 3).

Observe that constraints of the form $x \sim c$ with $\sim \in \{=, >, <\}$ allows to compare variables to absolute time values. A semi-simple periodicity constraint is a conjunction between a simple periodicity constraint and a conjunction of atomic constraints of the form $x \sim c$ with $\sim \in \{<, >, =\}$.

The interpretation of the constraints is the standard one. A valuation v is a map $v : \mathrm{VAR} \to \mathbb{Z}$. The satisfiability relation $v \models p$ is inductively defined in Figure 1.

- $v \models x \sim c \overset{\text{def}}{\Leftrightarrow} v(x) \sim c$ with $\sim \in \{<, >, =\}$; $v \models x = y \overset{\text{def}}{\Leftrightarrow} v(x) = v(y)$;
- $v \models x \equiv_k c \overset{\text{def}}{\Leftrightarrow} v(x)$ is equal to c modulo k;
- $v \models x \equiv_k y + c \overset{\text{def}}{\Leftrightarrow} v(x) - v(y)$ is equal to c modulo k;
- $v \models x \equiv_k y + [c_1, c_2] \overset{\text{def}}{\Leftrightarrow} v(x) - v(y)$ is equal to c modulo k for some $c_1 \leq c \leq c_2$;
- $v \models p \wedge p' \overset{\text{def}}{\Leftrightarrow} v \models p$ and $v \models p'$; $v \models \neg p \overset{\text{def}}{\Leftrightarrow}$ not $v \models p$;
- $v \models \exists\ x\ p \overset{\text{def}}{\Leftrightarrow}$ there is $c \in \mathbb{Z}$ such that $v[x \leftarrow c] \models p$ where $v[x \leftarrow c](x') = v(x')$ if $x' \neq x$, and $v[x \leftarrow c](x) = c$.

Fig. 1. Standard semantics for IPC^{++} constraints

It is worth observing that $x \equiv_k y + [c_1, c_2]$ is not symmetrical with respect to x and y. However, $y \equiv_k x + [c_1, c_2]$ is equivalent to $x \equiv_k y + [k - c_2, k - c_1]$.

Given p in IPC^{++} with free variables $x_1, \ldots, x_k$ (in the order of enumeration of the variables), $\mathrm{sol}(p)$ denotes the set of k-tuples $\langle z_1, \ldots, z_k \rangle \in \mathbb{Z}^k$ such that

$[x_1 \leftarrow z_1, \ldots, x_k \leftarrow z_k] \models p$. sol($p$) is a semilinear set of k-tuples since IPC^{++} is obviously a fragment of Presburger arithmetic [GS66]. Given a constraint language L, the L-satisfiability problem is to decide given a constraint $p \in$ L whether sol(p) is non-empty. Without any loss of generality, in the sequel we assume that p contains at least one free variable (otherwise consider $(x_1 \equiv_1 0) \wedge p$ and x_1 does not occur in p) and in p a variable cannot occur both free and bounded.

The expressive power of a constraint language L can be measured by the set $\{\mathrm{sol}(p) : p \in \mathrm{L}\}$. For instance, $\mathrm{IPC}^{\{\exists,<\}}$ is as expressive as IPC^{+} since $x \equiv_k y + [c_1, c_2]$ is equivalent to $\bigvee_{c_1 \leq c \leq c_2} x \equiv_k y + c$. However, because all the natural numbers are encoded in binary, IPC^{+} may be more concise than $\mathrm{IPC}^{\{\exists,<\}}$. The introduction of the concise atomic constraints of the form $x \equiv_k y + [c_1, c_2]$ is motivated by the existence of similar constraints in the calendar logic developed in [Ohl94].

2.2 Definition of $\mathrm{PLTL}^{\mathrm{mod}}$

The atomic formulae of $\mathrm{PLTL}^{\mathrm{mod}}$ are expressions of the form

$$p[x_1 \leftarrow \mathsf{X}^{i_1} x_{j_1}, \ldots, x_k \leftarrow \mathsf{X}^{i_k} x_{j_k}]$$

where p is a constraint of IPC^{++} with free variables $x_1, \ldots, x_k$ (in the order of enumeration of the variables) and $p[x_1 \leftarrow \mathsf{X}^{i_1} x_{j_1}, \ldots, x_k \leftarrow \mathsf{X}^{i_k} x_{j_k}]$ is obtained from p by replacing every occurrence of x_u by x_{j_u} preceded by i_u next symbols for $1 \leq u \leq k$. For instance, the formula $x \equiv_2 0 \wedge \Box(\mathsf{X}x \equiv_2 x + 1)$ states that the value of x is even on states of even indices. Otherwise stated, the atomic formulae of $\mathrm{PLTL}^{\mathrm{mod}}$ are the constraints of IPC^{++} except that the variables are of the form $\mathsf{X}^j x_i$.

The formulae of $\mathrm{PLTL}^{\mathrm{mod}}$ are defined by the following grammar:

$$\phi ::= p[x_1 \leftarrow \mathsf{X}^{i_1} x_{j_1}, \ldots, x_k \leftarrow \mathsf{X}^{i_k} x_{j_k}] \mid \neg\phi \mid \phi \wedge \phi \mid \mathsf{X}\phi \mid \phi \mathsf{U} \phi \mid \mathsf{X}^{-1}\phi \mid \phi \mathsf{S} \phi,$$

where p belongs to IPC^{++}. As usual, X is the next-time operator, X^{-1} is the previous past-time operator, U is the until operator, and S is the since past-time operator (see below the semantics). More generally, we write PLTL(L) to denote the variant of $\mathrm{PLTL}^{\mathrm{mod}}$ where the atomic formulae are built from the constraint language L. Hence, $\mathrm{PLTL}^{\mathrm{mod}}$ is simply $\mathrm{PLTL}(\mathrm{IPC}^{++})$. We write LTL(L) to denote the restriction of PLTL(L) to the future-time operators X and U. We include past-time operators to the logic in order to capture the conciseness of LTL with past considered in [CFP02,CFP04]. However, the addition of a finite amount of MSO-definable temporal operators still guarantees the (forthcoming) PSPACE upper bound thanks to [GK03].

A model σ for $\mathrm{PLTL}^{\mathrm{mod}}$ is an ω-sequence of valuations of the form $\sigma : \mathbb{N} \times \mathrm{VAR} \rightarrow \mathbb{Z}$. The satisfiability relation $\models$ is inductively defined in Figure 2.

A very important aspect of $\mathrm{PLTL}^{\mathrm{mod}}$ rests on the fact that the values of variables at different states can be compared. We use the standard abbreviations

- $\sigma, i \models p[x_1 \leftarrow \mathsf{X}^{i_1}x_{j_1}, \ldots, x_k \leftarrow \mathsf{X}^{i_k}x_{j_k}]$ iff $[x_1 \leftarrow \sigma(i+i_1, x_{j_1}), \ldots, x_k \leftarrow \sigma(i+i_k, x_{j_k})] \models p$ (the second occurrence of $\models$ denotes the satisfiability relation in IPC^{++});
- $\sigma, i \models \phi \wedge \phi'$ iff $\sigma, i \models \phi$ and $\sigma, i \models \phi'$; $\sigma, i \models \neg\phi$ iff not $\sigma, i \models \phi$;
- $\sigma, i \models \mathsf{X}\phi$ iff $\sigma, i+1 \models \phi$; $\sigma, i \models \mathsf{X}^{-1}\phi$ iff $i > 0$ and $\sigma, i-1 \models \phi$;
- $\sigma, i \models \phi \mathsf{U} \phi'$ iff there is $j \geq i$ such that $\sigma, j \models \phi'$ and for every $i \leq k < j$, $\sigma, k \models \phi$;
- $\sigma, i \models \phi \mathsf{S} \phi'$ iff there is $0 \leq j \leq i$ such that $\sigma, j \models \phi'$ and for every $j < k \leq i$, $\sigma, k \models \phi$.

Fig. 2. Semantics for PLTL$^{\mathrm{mod}}$ formulae

$\Diamond\phi$, $\mathsf{F}^{-1}\phi$, $\Box\phi$, ... The satisfiability problem for PLTL$^{\mathrm{mod}}$ is to decide given a formula ϕ whether there is σ such that $\sigma, 0 \models \phi$. It is worth observing that adding to IPC^{++} constraints of one of the forms below leads to undecidability of the satisfiability problem of the corresponding extension of PLTL$^{\mathrm{mod}}$:

- $x = y + 1$ with $x, y \in$ VAR, see e.g. [CC00];
- $x < y$ with $x, y \in$ VAR;
- $x - y \geq c$ with $x, y \in$ VAR and $c \in \mathbb{N} \setminus \{0\}$.

A few other remarks are in order. No propositional variables are part of PLTL$^{\mathrm{mod}}$ but they can be easily simulated, for instance each P_i is encoded by $x_i = 1$ if x_i is not used for other purposes in the formula. When complexity issues are considered, all the integers are encoded in binary representation.

Observe that because of the presence of the past-time operator X^{-1}, we can also simulate the access to past values of variables (which we would write $\mathsf{X}^{-n}x$ for instance if $\overbrace{\mathsf{X}^{-1} \ldots \mathsf{X}^{-1}}^{n \text{ times}} \top$ holds true). Typically, $\mathsf{X}^{-2}x = x$ can be concisely translated into $\mathsf{X}^{-1}\mathsf{X}^{-1}\top \wedge \mathsf{X}^{-1}\mathsf{X}^{-1}(x = \mathsf{X}^2 x)$ assuming that if $\mathsf{X}^{-2}x$ is undefined, then the atomic constraint is interpreted by false.

Example 1. PLTL$^{\mathrm{mod}}$ is a quite rich and concise language. Formulae of PLTL$^{\mathrm{mod}}$ encode

- calendars and slices from [NS92];
- formulae of the form $[\tau]\phi$ from [Ohl94] where $[\tau]\phi$ is interpreted by "for every point of the interval τ, the formula ϕ holds" can be encoded by $\Box(\tau' \Rightarrow \phi)$. Here τ' is a constraint in IPC^{++} encoding τ. Unlike what is done in [Ohl94], no exponential-time reduction to propositional calculus (PC) is performed. Our logic PLTL$^{\mathrm{mod}}$ allows more efficient reasoning than an expensive translation into PC (see details in Sect. 4).

2.3 PLTL$^{\mathrm{mod}}$ as a concise calendar logic

In [CFP02], LTL with past is used for reasoning about calendars based on consistent granularities: this amounts to require that the truth values of propositional variables along the time line encode consistent granularities. As a major drawback, the encoding a period of n units requires a formula of size $\mathcal{O}(n)$ whereas the formula $\Box(\mathsf{X}x \equiv_n x+1)$ in PLTL$^{\mathrm{mod}}$ does the job with only $\mathcal{O}(log(n))$ symbols (remember that we encode the integers with a binary representation). A similar blow up occurs in the translation of pure Calendar Logic [Ohl94] into PC with an exponential increase of the size of formulae, which leads to a decision procedure in double exponential-time (to be compared with our PSPACE decision procedure in Sect. 4). Other advantages of our formalism in comparison with [Ohl94,CFP02] is that we specify in the logical language the granularities.

By way of example, we provide a partial encoding of Gregorian calendar with PLTL$^{\mathrm{mod}}$ formulae. Because of lack of space, we do not provide all the details.

- $sec \equiv_{60} 0 \wedge \Box(\mathsf{X}sec \equiv_{60} sec + 1 \wedge 0 \leq sec < 60)$; the second is the time unit (granularity);
- $min \equiv_{60} 0 \wedge \Box(0 \leq min < 60) \wedge$
  $\Box((sec = 59 \Rightarrow \mathsf{X}min \equiv_{60} min + 1) \wedge (sec \neq 59 \Rightarrow \mathsf{X}min = min))$;
- $hour \equiv_{60} 0 \wedge \Box(0 \leq hour < 24) \wedge$
  $\Box((min = 59 \wedge sec = 59 \Rightarrow \mathsf{X}hour \equiv_{24} hour + 1) \wedge (min \neq 59 \vee sec \neq 59 \Rightarrow$ $\mathsf{X}hour = hour))$;
- $day \equiv_7 0 \wedge \Box(0 \leq day < 7) \wedge$
  $\Box((min = 59 \wedge sec = 59 \wedge hour = 23 \Rightarrow \mathsf{X}day \equiv_7 day + 1) \wedge$
  $\Box((min \neq 59 \vee sec \neq 59 \vee hour \neq 23 \Rightarrow \mathsf{X}day = day)$;
- similarly, one can defined *day-in-month*, *month*, and *year* assuming that there is some end dates. In many practical situations, the problem of infiniteness can be circumverted by fixing some end date far ahead. Indeed, with PLTL$^{\mathrm{mod}}$ formalism this can be done concisely, for instance $x < 2^n$ requires only $\mathcal{O}(n)$ symbols since the integers are encoded with a binary representation.
- the sentence "ϕ holds sometime next monday" (where the time unit is the second) is encoded by the following conjunction (depending whether the current day is monday or not):

$$\overbrace{p \Rightarrow p\mathsf{U}((\neg p \wedge (\neg p\mathsf{U}(p\mathsf{U}(p \wedge \phi))))))}^{\text{current day is monday}} \wedge \overbrace{\neg p \Rightarrow (\neg p\mathsf{U}(p\mathsf{U}(p \wedge \phi)))}^{\text{current day is not monday}}$$

  with $p = day \equiv_7 0$. It is easy to see that this can be easily generalized to any interval of time (next year, next week, previous year, etc.).

2.4 Model-Checking

The languages of the form PLTL(L) are of course well-designed to perform model-checking of counter automata, similarly to what is done in [Čer94,DD03] and [Rev02, Chap. 6]. Given a constraint language L, a PLTL(L)-automaton is a Büchi automaton $\mathcal{A}$ over the alphabet of PLTL(L) formulas: transitions are of the form $q \xrightarrow{\phi} q'$. To each ω-word $w = \phi_0\phi_1\cdots$ accepted by $\mathcal{A}$, we associate a PLTL$^{\mathrm{mod}}$ model σ which satisfies $\sigma, i \models \phi_i$ for $i \geq 0$. Let $l(\mathcal{A})$ denote the set

$l(\mathcal{A}) = \{\sigma : \mathbb{N} \times \mathrm{VAR} \to \mathbb{Z} \mid \exists w$ accepted by $\mathcal{A}$ such that $\sigma, i \models w(i)$ for each $i\}$. The model-checking problem for PLTL(L) is defined as follows: given a PLTL(L)-automaton $\mathcal{A}$ and a PLTL(L) formula ϕ, is there a $\sigma \in l(\mathcal{A})$ such that $\sigma \models \phi$? (in symbols $\mathcal{A} \models_{\exists} \phi$?) A natural relaxed version of the problem consists in restricting the labels on transitions to Boolean combinations of PLTL(L) atomic formulae.

Theorem 1. *The model-checking and satisfiability problems for* $\mathrm{PLTL}^{\mathrm{mod}}$ *are inter-reducible with respect to logspace transformations.*

The proof is similar to the proof of [DD03, Theorem 8.3] (which is itself based on a proof in [SC85]) and is omitted here. In Sect. 6 we shall show how $\mathrm{LTL}(\mathrm{IPC}^{++})$-automata naturally encode extended single-string automata from [LM01, Sect. 5].

In the sequel, only satisfiability problems are explicitly treated.

3 First-order theory of integer periodicity constraints

Given p in IPC^{++} with free variables $x_1, \ldots, x_k$, we shall construct a finite partition of $\mathbb{Z}^k$ such that

- every region can be represented by a semi-simple periodicity constraint;
- for all k-tuples $\overline{z}$ and $\overline{z'}$ in a given region of the partition, $\overline{z} \in \mathrm{sol}(p)$ iff $\overline{z'} \in \mathrm{sol}(p)$.

In this way, we shall be able to finitely represent the set of solutions $\mathrm{sol}(p)$ and such a representation will be easy to manipulate since it can be viewed as a disjunction of semi-simple periodicity constraints. This is actually a standard requirement when an infinite set of tuples has to be finitely abstracted, see e.g. the clock regions for timed automata in [AD94], the quantifier elimination procedure for discrete point constraint language in [Kou94] and the reducibility of extended single-string automata in [LM01], to quote a few examples (see also the symbolic transition systems of the class one in [HMR03]).

3.1 Quantifier elimination

Quantifier elimination (QE) is a known method to show decidability of logical theories, see e.g. [Pre29,Tar51,KK67,FR75]. In this section, we establish such a property to prove the PSPACE upper bound of the IPC^{++}-satisfiability problem. Let p be a constraint in IPC^{++} such that

- $c_1 < \ldots < c_n$ are the constants in p occurring in constraints of the form $x \sim c$ with $\sim \in \{<, >, =\}$; we fix $c_0 = -\infty$ and $c_{n+1} = +\infty$, this is done to simplify the notations in some places;
- $k_1, \ldots, k_u$ are the natural numbers occurring in constraints of the form $x \equiv_k y + [d_1, d_2]$; we fix K to be the least common multiple of $1, k_1, \ldots, k_u$, denoted by $lcm(1, k_1, \ldots, k_u)$. K is in $2^{\mathcal{O}(|p|)}$.

We define from p an equivalence relation $\sim_p$ between elements of $\mathbb{Z}$ as follows: $z \sim_p z' \overset{\text{def}}{\Leftrightarrow}$

1. for all $i \leq j \in \{0, \ldots, n+1\}$, $c_i \leq z \leq c_j$ iff $c_i \leq z' \leq c_j$;
2. for every $l \in \{0, \ldots, K-1\}$, $z \equiv_K l$ iff $z' \equiv_K l$.

Hence, the number of equivalence classes of $\sim_p$ is bounded by $(n+1) \times K$, that is in $\mathcal{O}(2^{|p|})$. The idea behind the definition of $\sim_p$ is simply that $z \sim_p z'$ iff z and z' cannot be distinguished by constraints of IPC$^+$ that use only $c_1, \ldots, c_n$ and $k_1, \ldots, k_u$. For instance, it is easy to check that for every $j \in \{1, \ldots, n\}$, $\{c_i\}$ is an equivalence class of $\sim_p$. The relation $\sim_p$ extended to tuples will not be a simple component-wise extension because of the presence of equality in IPC^{++}. For $k \geq 1$, we say that $\langle z_1, \ldots, z_k \rangle = \overline{z} \sim_p^k \overline{z'} = \langle z'_1, \ldots, z'_k \rangle$ iff

- for every $i \in \{1, \ldots, k\}$, $z_i \sim_p z'_i$;
- for all $i, j \in \{1, \ldots, k\}$, $z_i = z_j$ iff $z'_i = z'_j$.

If $x_1, \ldots, x_k$ are the free variables in p, we write $\overline{z} \sim_p \overline{z'}$ instead of $\overline{z} \sim_p^k \overline{z'}$. The number of equivalence classes of $\sim_p$ (on k-tuples) is bounded by $(n+1) \times K \times 2^{k^2}$.

In order to prove Lemma 1 below, we introduce the following notation. Given $1 \leq i_1 < \ldots < i_l \leq k$, we write $\langle z_1, \ldots, z_k \rangle^{\{i_1, \ldots, i_l\}}$ to denote the subsequence $\langle z_{i_1}, \ldots, z_{i_l} \rangle$.

Lemma 1. *Let p be a constraint in* IPC^{++} *with k free variables and $\overline{z}, \overline{z'} \in \mathbb{Z}^k$. $\overline{z} \in \mathrm{sol}(p)$ and $\overline{z} \sim_p \overline{z'}$ imply $\overline{z'} \in \mathrm{sol}(p)$.*

Proof. By induction on the size of p. The base cases are by an easy verification. Let us consider the different cases of the induction step.
Case 1: $p = p_1 \wedge p_2$.
Suppose $\langle z_1, \ldots, z_k \rangle = \overline{z} \in \mathrm{sol}(p)$ and $\overline{z} \sim_p \overline{z'} = \langle z'_1, \ldots, z'_k \rangle$, that is $[x_1 \leftarrow z_1, \ldots, x_k \leftarrow z_k] \models p$. Let $x_{i_1}, \ldots, x_{i_s}$ be the variables occurring in p_1 and $x_{j_1}, \ldots, x_{j_l}$ be the variables occurring in p_2. For $\alpha \in \{1, 2\}$ and $m, m' \in \mathbb{Z}$, one can check that $m \sim_p m'$ implies $m \sim_{p_\alpha} m'$ since obviously all the constants occurring in p_α occurs in p. Moreover, one can verify that $\overline{z}^{\{i_1, \ldots, i_s\}} \sim_{p_1} \overline{z'}^{\{i_1, \ldots, i_s\}}$ and $\overline{z}^{\{j_1, \ldots, j_l\}} \sim_{p_2} \overline{z'}^{\{j_1, \ldots, j_l\}}$. By the induction hypothesis, $[x_{i_1} \leftarrow z'_{i_1}, \ldots, x_{i_s} \leftarrow z'_{i_s}] \models p_1$ and $[x_{j_1} \leftarrow z'_{j_1}, \ldots, x_{j_l} \leftarrow z'_{j_l}] \models p_2$. Since for the variables x in $\{x_{i_1}, \ldots, x_{i_s}\} \cap \{x_{j_1}, \ldots, x_{i_l}\}$, $[x_{i_1} \leftarrow z'_{i_1}, \ldots, x_{i_s} \leftarrow z'_{i_s}](x) = [x_{j_1} \leftarrow z'_{j_1}, \ldots, x_{j_l} \leftarrow z'_{j_l}](x)$, we obtain that $[x_1 \leftarrow z'_1, \ldots, x_k \leftarrow z'_k] \models p$.
Case 2: $p = \neg p_1$.
Suppose $\overline{z} \in \mathrm{sol}(p)$ and $\overline{z} \sim_p \overline{z'}$. Hence $\overline{z} \notin \mathrm{sol}(p_1)$. By the induction hypothesis, $\overline{z'} \notin \mathrm{sol}(p_1)$, whence $\overline{z'} \in \mathrm{sol}(p)$.
Case 3: $p = \exists\ x_{k+1}\ p_1$.
Suppose $\langle z_1, \ldots, z_k \rangle = \overline{z} \in \mathrm{sol}(p)$ and $\overline{z} \sim_p \overline{z'} = \langle z'_1, \ldots, z'_k \rangle$, that is $[x_1 \leftarrow z_1, \ldots, x_k \leftarrow z_k] \models p$. By definition of the satisfiability relation $\models$, there is $z_{k+1} \in \mathbb{Z}$ such that $[x_1 \leftarrow z_1, \ldots, x_k \leftarrow z_k, x_{k+1} \leftarrow z_{k+1}] \models p_1$. However, $\overline{z}, z_{k+1} \sim_{p_1} \overline{z'}, z_{k+1}$ and by the induction hypothesis, $[x_1 \leftarrow z'_1, \ldots, x_k \leftarrow z'_k, x_{k+1} \leftarrow z_{k+1}] \models p_1$. By definition of the satisfiability relation $\models$, $[x_1 \leftarrow z'_1, \ldots, x_k \leftarrow z'_k] \models p$.

Each equivalence class of $\sim_p$ on $\mathbb{Z}$ can be represented by a triple $\langle i,j,l\rangle$ with $i,j \in \{0,\ldots,n+1\}$ and $l \in \{0,\ldots,K-1\}$ such that

- $i \leq j \leq i+1$;
- if $i=j$ and $c_i \equiv_K l$ then $\langle i,j,l\rangle$ represents the equivalence class $\{c_i\}$;
- if $j=i+1$, then $\langle i,j,l\rangle$ represents the equivalence class $\{z \in \mathbb{Z} : c_i < z < c_{i+1},\ \text{and}\ z \equiv_K l\}$ if this set is non empty.

We introduce the map $[\cdot] : \mathbb{Z} \to \{0,\ldots,n+1\}^2 \times \{0,\ldots,K-1\}$ such that $[z]$ is the equivalence class of $\sim_p$ containing z. For instance, if $c_i \equiv_K 0$, then $[c_i] = \langle i,i,0\rangle$. By extension, given Y a non-empty finite subset of $\mathbb{N}$ of cardinality k representing a set of variable indices, we introduce the map

$$[\cdot]^Y : \mathbb{Z}^k \to (\{0,\ldots,n+1\}^2 \times \{0,\ldots,K-1\})^k \times \mathcal{P}(Y^2)$$

such that

$$[\langle z_1,\ldots,z_k\rangle]^Y = \langle\langle [z_1],\ldots,[z_k]\rangle, \{\langle J_i,J_j\rangle \in Y^2 : z_i = z_j\}\rangle,$$

where $Y = \{J_1,\ldots,J_k\}$ and $J_1 < \ldots < J_k$. If p has free variables $x_1,\ldots,x_k$, the finite set $(\{0,\ldots,n+1\}^2 \times \{0,\ldots,K-1\})^k \times \mathcal{P}(\{1,\ldots,k\}^2)$ will represent the equivalence classes of $\sim_p$ on k-tuples.

The proof of Lemma 2 below is by an easy verification.

Lemma 2. *Let $Y = \{J_1,\ldots,J_k\}$ be a non-empty finite subset of $\mathbb{N}$ with $J_1 < \ldots < J_k$. Checking whether $u \in (\{0,\ldots,n+1\}^2 \times \{0,\ldots,K-1\})^k \times \mathcal{P}(Y^2)$ belongs to the image of $[\cdot]^Y$ can be checked in polynomial-time in $|p| + |Y|$.*

In Lemma 2 above, $|u|$ is of polynomial size in $|p| + |Y|$ with $|Y| = \Sigma_{i=1}^{k}(1 + log(J_i))$.

If p contains k free variables $x_1,\ldots,x_k$, we write D_p to denote the domain $(\{0,\ldots,n+1\}^2 \times \{0,\ldots,K-1\})^k \times \mathcal{P}(\{1,\ldots,k\}^2)$ and D_p^{sat} to denote the set $\{[\overline{z}]^{\{1,\ldots,k\}} \in D_p : \overline{z} \in \mathrm{sol}(p)\}$. The set D_p is indeed a finite abstraction of the infinite domain $\mathbb{Z}^k$ with respect to the constraint p and D_p^{sat} is a finite representation of the possibly infinite set $\mathrm{sol}(p)$. In the sequel, we show how an element of D_p^{sat} can be represented by a semi-simple periodicity constraint.

To each $\langle i,j,l\rangle \in \{0,\ldots,n+1\}^2 \times \{0,\ldots,K-1\}$, and variable index $\alpha \in \mathbb{N}$, we associate a semi-simple periodicity constraint $\mathrm{IPC}^{<}(\langle i,j,l\rangle,\alpha)$ in $\mathrm{IPC}^{\{<\}}$ with free variable x_α defined as follows:

$$\mathrm{IPC}^{<}(\langle i,j,l\rangle,\alpha) = (x_\alpha \equiv_K l) \wedge \begin{cases} x_\alpha = c_i \text{ if } i = j, \\ (c_i < x_\alpha) \wedge (x_\alpha < c_j) \\ \text{if } j = i+1,\ i \neq 0, \text{ and } j \neq n+1, \\ x_\alpha < c_1 \text{ if } i = 0 \text{ and } j = 1, \\ c_n < x_\alpha \text{ if } i = n \text{ and } j = n+1, \\ \text{undefined otherwise.} \end{cases}$$

The following lemma (not difficult to show) makes explicit the relationship between the constraints generated by the map $\mathrm{IPC}^{<}$ and the map $[\cdot]$.

Lemma 3. *For all $z \in \mathbb{Z}$ and $\langle i,j,l\rangle \in \{0,\ldots,n+1\}^2 \times \{0,\ldots,K-1\}$, we have $[x_\alpha \leftarrow z] \models \mathrm{IPC}^<(\langle i,j,l\rangle,\alpha)$ iff $[z] = \langle i,j,l\rangle$.*

We are now able to show that IPC^{++} satisfies (QE) by appropriately extending the map $\mathrm{IPC}^<$. To each $\langle\langle t_1,\ldots,t_k\rangle, X\rangle \in D_p$ we associate a semi-simple periodicity constraints $\mathrm{IPC}^{++}(\langle\langle t_1,\ldots,t_k\rangle, X\rangle)$ defined by

$$(\bigwedge_{1\leq i\leq k} \mathrm{IPC}^<(t_i,i)) \wedge (\bigwedge_{\langle i,j\rangle\in X} x_i = x_j) \wedge (\bigwedge_{\langle i,j\rangle\notin X} \neg(x_i = x_j)).$$

The following lemma (also not difficult to show) makes explicit the relationship between the constraints generated by the map $\mathrm{IPC}^{++}(\cdot)$ and the map $[\cdot]^{\{1,\ldots,k\}}$.

Lemma 4. *For all $\langle z_1,\ldots,z_k\rangle \in \mathbb{Z}^k$ and $u \in D_p$, we have $[x_1 \leftarrow z_1,\ldots,x_k \leftarrow z_k] \models \mathrm{IPC}^{++}(u)$ iff $[\langle z_1,\ldots,z_k\rangle]^{\{1,\ldots,k\}} = u$.*

Theorem 2. IPC^{++} *admits quantifier elimination.*

Proof. Let p be a constraint in IPC^{++} with free variables $x_1,\ldots,x_k$. We define below a constraint p' in IPC^{++} such that $\mathrm{sol}(p) = \mathrm{sol}(p')$:

$$p' = \bigvee_{\langle\langle t_1,\ldots,t_k\rangle,X\rangle\in D_p^{sat}} \mathrm{IPC}^{++}(\langle\langle t_1,\ldots,t_k\rangle, X\rangle).$$

Equality between $\mathrm{sol}(p)$ and $\mathrm{sol}(p')$ can be proved by using Lemma 4.

3.2 PSPACE-complete satisfiability problem

We establish that IPC^{++}-satisfiability is decidable in polynomial space.

Theorem 3. IPC^{++}*-satisfiability is* PSPACE*-complete.*

Proof. PSPACE-hardness is immediate by reducing QBF. We omit the details of this obvious reduction. Satisfiability in PSPACE can be shown via a procedure similar to first-order model-checking [CM77], details are given below.

First, some preliminary definitions. Given a sequence $\langle s_1,\ldots,s_k\rangle$, we write $\langle s_1,\ldots,s_k\rangle[i_1 \leftarrow t_1,\ldots,i_s \leftarrow t_s]$ to denote the sequence obtained from $\langle s_1,\ldots,s_k\rangle$ by replacing s_{i_j} by t_j for every $j \in \{1,\ldots,s\}$. We shall define a function $\mathrm{SAT}(p)$ that checks satisfiability of the constraint p in IPC^{++}. To do so, we introduce an auxiliary function MC which is indeed the core of our procedure. Let p be a constraint in IPC^{++} with occurrences of the variables $x_1,\ldots,x_k$. The free variables of p are $x_{i_1},\ldots,x_{i_s}$ with $1 \leq i_1 < i_2 < \ldots < i_s \leq k$. The function SAT is defined in Figure 3.

Observe that condition 2. in the definition of SAT can be checked in polynomial-time in $|p|$ by Lemma 2. Moreover, it will not be difficult to show that MC (defined below) runs in PSPACE: polynomial recursion depth and quantification over exponential size sets (which requires only polynomial space) guarantees this upper bound. MC has four arguments:

SAT(p):

if there is $\langle\langle t_1, \ldots, t_s\rangle, X\rangle \in (\{0, \ldots, n+1\}^2 \times \{0, \ldots, K-1\})^s \times \mathcal{P}(\{i_1, \ldots, i_s\}^2)$ such that
1. $\langle\langle t_1, \ldots, t_s\rangle, X\rangle$ belongs to the image of $[\cdot]^{\{i_1,\ldots,i_s\}}$ and
2. MC(p, p, $\langle\overbrace{\langle -, \ldots, - \rangle}^{k \text{ times}}[i_1 \leftarrow t_1, \ldots, i_s \leftarrow t_s], X\rangle$, $\{i_1, \ldots, i_s\}$) returns "Yes";

then return "Yes" otherwise return "No".

Fig. 3. Function SAT

1. a constraint p of IPC^{++};
2. a subconstraint p' of p;
3. an interpretation of the free variables of p' represented in an abstract fashion with the use of a padding symbol '$-$';
4. the set of indices of the free variables of p'.

There is certainly a bit of redundancy in the arguments: the positions of the third argument with values different from the padding symbol '$-$' are precisely the elements of the fourth argument. However, this is not crucial for the result we want to establish. MC is indeed a model-checking procedure since the third argument provides an interpretation for the free variables of the second argument. MC returns "Yes" iff this interpretation forces the second argument to hold true. The function MC is defined by a simple case analysis as shown in Figure 4.

In the case $p' = \exists x_i\ p''$, condition 1. can be checked in polynomial-time in $|p|$.

The PSPACE-completeness of IPC^{++}-satisfiability does not play in favor of the tractability of this first-order theory, especially if one compares it with NLOGSPACE consistency problems. However, Presburger arithmetic is of much higher complexity [FR74,Ber80] and PSPACE-hardness is the optimal lower bound one can expect for PSPACE-complete PLTL over fragments of Presburger arithmetic.

Corollary 1. *Let p be a constraint in* IPC^{++}*. Checking whether $u \in D_p$ belongs to D_p^{sat} can be done in* PSPACE*.*

Proof. Let $u \in D_p$. One can show that $u \in D_p^{sat}$ iff $p \wedge \text{IPC}^{++}(u)$ is satisfiable. Hence, the PSPACE upper bound.

Observe also that IPC$^{++}(\langle\langle t_1, \ldots, t_k\rangle, X\rangle)$ is indeed a set of signed atomic constraints of the form $s\ (x_i = x_j)$, $s\ (x_i \sim c_j)$, and $s\ (x_i \equiv_K l)$ with the sign s in $\{\epsilon, \neg\}$ and $\sim \in \{<, >, =\}$.

The problem described in Corollary 1 is actually a model-checking problem (easily solvable with the procedure MC) where the interpretation of the variables

MC(p, p', $\langle\langle s_1,\ldots,s_k\rangle, X\rangle$, Y):

- if $p' =$ "$x_i = x_j$" and $\langle i,j\rangle \in X$ then return "Yes";
- if $p' =$ "$x_i \equiv_\alpha x_j + [c_1, c_2]$" and there is $c \in [c_1, c_2]$ such that $l_i \equiv_\alpha l_j + c$ then return "Yes" ($s_i = \langle min_i, max_i, l_i\rangle$ and $s_j = \langle min_j, max_j, l_j\rangle$);
- the other base cases from the atomic formulae of IPC^{++} are treated in a similar fashion and one can check that this requires only polynomial-time in p;
- in the case $p' = p_1 \wedge p_2$, let $x_{i_1},\ldots,x_{i_\alpha}$ be the free variables of p_1 and $x_{j_1},\ldots,x_{j_\beta}$ be the free variables of p_2. If the two calls below returns "Yes"
  1. MC(p, p_1, $\langle\langle -,\ldots,-\rangle[i_1 \leftarrow s_{i_1},\ldots,i_\alpha \leftarrow s_{i_\alpha}], X \cap \{i_1,\ldots,i_\alpha\}^2\rangle$, $\{i_1,\ldots,i_\alpha\}$)
  2. MC(p, p_2, $\langle\langle -,\ldots,-\rangle[j_1 \leftarrow s_{j_1},\ldots,j_\beta \leftarrow s_{j_\beta}], X \cap \{j_1,\ldots,j_\beta\}^2\rangle$, $\{j_1,\ldots,j_\beta\}$)

  then return "Yes".
- in the case $p' = \neg p''$, if MC(p, p'', $\langle\langle s_1,\ldots,s_k\rangle, X\rangle$, Y) returns "No", then return "Yes";
- if $p' = \exists x_i\ p''$ then if there are $t \in \{0,\ldots,n+1\}^2 \times \{0,\ldots,K-1\}$ and $X \subseteq X' \subseteq (Y \cup \{i\})^2$ such that
  1. $\langle(\langle s_1,\ldots,s_k\rangle[i \leftarrow t])^{Y\cup\{i\}}, X'\rangle$ belongs to the image of $[\cdot]^{Y\cup\{i\}}$;
  2. MC(p, p'', $\langle\langle s_1,\ldots,s_k\rangle[i \leftarrow t], X'\rangle$, $Y \cup \{i\}$) returns "Yes";

  then return "Yes".

Return "No".

Fig. 4. Function MC

is done modulo the equivalence classes of $\sim_p$. By Lemma 1, this reasoning modulo is sufficient.

Corollary 2. *Given a constraint p in* IPC^{++}*, one can compute an equivalent quantifier-free p' in polynomial space in $|p|$ (but $|p'|$ is in $\mathcal{O}(2^{|p|})$).*

This is a mere consequence of the proof of Theorem 2, Corollary 1, and the fact that all the elements of D_p can be enumerated using only polynomial space in $|p|$.

4 Complexity of PLTL$^{\mathrm{mod}}$

Let ϕ be a PLTL$^{\mathrm{mod}}$ formula with

- free variables $x_1,\ldots,x_s$;
- constants $c_1 < \ldots < c_n$ ($c_0 = -\infty$ and $c_{n+1} = +\infty$);
- natural numbers $k_1,\ldots,k_u$ occurring in the context of $\equiv$-atomic formulae and their lcm is denoted by K.

Without any loss of generality, we can assume that these sets of integers/variables are non-empty. Let $\mathsf{X}(\phi)$ be one plus the greatest i with some term $\mathsf{X}^i x_j$ occurring in ϕ. For instance, $\mathsf{X}(\phi)$ with $\phi = \Box(\mathsf{X}x \equiv_n x+1)$ is equal to two. In the sequel, we pose $l = \mathsf{X}(\phi)$. l is the maximal number of consecutive states necessary to evaluate an atomic subformula of ϕ.

We shall provide in the sequel a procedure to decide satisfiability of ϕ using polynomial space in $|\phi|$.

4.1 Abstraction of PLTL$^{\mathbf{mod}}$ models

By definition, a model σ of ϕ is a structure $\sigma : \mathbb{N} \times \{x_1, \ldots, x_s\} \to \mathbb{Z}$ such that $\sigma, 0 \models \phi$. However, each $\sigma(i) : \{x_1, \ldots, x_s\} \to \mathbb{Z}$ can take an infinite amount of values. By contrast, for classical LTL, there is a finite amount of interpretations over a finite set of propositional variables. That is why we shall abstract such valuations as elements of a finite set, more precisely as elements of the set

$$(\{0, \ldots, n+1\}^2 \times \{0, \ldots, K-1\})^k \times \mathcal{P}(\{1, \ldots, k\}^2)$$

with $k = s \times l$. This provides evidence that PLTL(IPC^{++})-automata are in the class one of symbolic transitions systems following the classification in [HMR03]. The rest of this section is dedicated to the construction of such abstractions by using Sect. 3.

Another way to understand a structure $\sigma : \mathbb{N} \times \{x_1, \ldots, x_s\} \to \mathbb{Z}$ with the PLTL$^{\mathrm{mod}}$ semantics, is to view it as a structure $\sigma' : \mathbb{N} \times (\{x_1, \ldots, x_s\} \times \{0, \ldots, l-1\}) \to \mathbb{Z}$ such that

(C1) for all $i \in \mathbb{N}$, $\alpha \in \{1, \ldots, s\}$, and $\beta \in \{1, \ldots, l-1\}$, $\sigma'(i, \langle x_\alpha, \beta \rangle) = \sigma'(i+1, \langle x_\alpha, \beta - 1 \rangle)$.

In that way, the pair $\langle x_\alpha, \beta \rangle$ plays the rôle of the term $\mathsf{X}^\beta x_\alpha$. So far, the profile of σ' depends on ϕ by the value l and by the number of variables s but one has also to relate σ' with σ. The condition (C2) below does the job:

(C2) for all $i \in \mathbb{N}$ and $\alpha \in \{1, \ldots, s\}$, $\sigma'(i, \langle x_j, 0 \rangle) = \sigma(i, x_j)$.

The following lemma is now easy to establish.

Lemma 5.

(I) *Given $\sigma : \mathbb{N} \times \{x_1, \ldots, x_s\} \to \mathbb{Z}$, there is a unique $\sigma' : \mathbb{N} \times (\{x_1, \ldots, x_s\} \times \{0, \ldots, l-1\}) \to \mathbb{Z}$ satisfying (C1) and (C2).*

(II) *Given $\sigma' : \mathbb{N} \times (\{x_1, \ldots, x_s\} \times \{0, \ldots, l-1\}) \to \mathbb{Z}$ satisfying (C1), there is a unique $\sigma : \mathbb{N} \times \{x_1, \ldots, x_s\} \to \mathbb{Z}$ satisfying (C2).*

As a corollary:

Lemma 6. *ϕ is satisfiable iff there is a structure $\sigma' : \mathbb{N} \times (\{x_1, \ldots, x_s\} \times \{0, \ldots, l-1\}) \to \mathbb{Z}$ satisfying (C1) such that $\sigma', 0 \models \phi'$ where ϕ' is obtained from ϕ by replacing every occurrence of $\mathsf{X}^\beta x_\alpha$ by $\langle x_\alpha, \beta \rangle$.*

In Lemma 6 above, we assume that $\sigma', i \models p[x_1 \leftarrow \langle x_{j_1}, \beta_1\rangle, \ldots, x_d \leftarrow \langle x_{j_d}, \beta_d\rangle]$ holds true with $p \in \mathrm{IPC}^{++}$ and p has free variables $x_1, \ldots, x_d$ whenever $[x_1 \leftarrow \sigma'(i, \langle x_{j_1}, \beta_1\rangle), \ldots, x_d \leftarrow \sigma'(i, \langle x_{j_d}, \beta_d\rangle)] \models p$ in IPC^{++}. For the Boolean and temporal operators, the satisfiability relation $\models$ on structures σ' is defined in the obvious homomorphic way.

Let us now abstract the structures of the form $\sigma' : \mathbb{N} \times (\{x_1, \ldots, x_s\} \times \{0, \ldots, l-1\}) \rightarrow \mathbb{Z}$. We pose $k = s \times l$ and we write D_ϕ to denote the set $(\{0, \ldots, n+1\}^2 \times \{0, \ldots, K-1\})^k \times \mathcal{P}(\{1, \ldots, k\}^2)$ by similarity to the developments made in Sect. 3. D_ϕ^{sat} is defined as the subset of D_ϕ which is the image of $[\cdot]^{\{1,\ldots,k\}}$. In order to relate terms of the form $\mathsf{X}^\beta x_\alpha$ and variables x_i ($i \in \{1, \ldots, k\}$), we introduce the map $f : \{x_1, \ldots, x_s\} \times \{0, \ldots, l-1\} \rightarrow \{1, \ldots, k\}$ as the bijection defined by $f(\langle x_\alpha, \beta\rangle) = s \times \beta + \alpha$. The inverse function f^{-1} can be easily defined with the operations of the Euclidean division. Details are omitted here. One can check that $f^{-1}(1), f^{-1}(2), \ldots, f^{-1}(k)$ is precisely the sequence

$$\langle x_1, 0\rangle, \langle x_2, 0\rangle, \ldots, \langle x_s, 0\rangle, \langle x_1, 1\rangle, \ldots, \langle x_1, l-1\rangle, \langle x_2, l-1\rangle, \ldots, \langle x_s, l-1\rangle,$$

that is, first the variables at the current state are enumerated, then the variables at the next state are enumerated and so on.

Another way to understand a structure $\sigma : \mathbb{N} \times (\{x_1, \ldots, x_s\} \times \{0, \ldots, l-1\}) \rightarrow \mathbb{Z}$ is to view it as a structure $\sigma' : \mathbb{N} \rightarrow D_\phi^{\mathrm{sat}}$ such that

(C3) for every $i \in \mathbb{N}$, if $\sigma'(i) = \langle\langle t_1, \ldots, t_k\rangle, X\rangle$ and $\sigma'(i+1) = \langle\langle t'_1, \ldots, t'_k\rangle, X'\rangle$ then
1. $\langle t_{s+1}, \ldots, t_k\rangle = \langle t'_1, \ldots, t'_{k-s}\rangle$ (shift of the values of s first variables)
2. $X \cap \{s+1, \ldots, k\}^2 = \{\langle u+s, v+s\rangle : \langle u, v\rangle \in X',\ u+s \leq k,\ v+s \leq k\}$ (preservation in X' of X restricted to the indices in $\{s+1, \ldots, k\}$).

One has also to relate σ' with σ. The condition (C4) below does the job. First we need a preliminary definition. Given $g : \{x_1, \ldots, x_s\} \times \{0, \ldots, l-1\} \rightarrow \mathbb{Z}$, we write g^k to denote the k-tuple $\langle g(f^{-1}(1)), \ldots, g(f^{-1}(k))\rangle$. g^k is simply a representation of g as a k-tuple of $\mathbb{Z}^k$ with $k = s \times l$.

(C4) for all $i \in \mathbb{N}$, $\sigma'(i) = [\sigma(i)^k]^{\{1,\ldots,k\}}$.

The following lemma can be established.

Lemma 7.

(I) *Given $\sigma : \mathbb{N} \times (\{x_1, \ldots, x_s\} \times \{0, \ldots, l-1\}) \rightarrow \mathbb{Z}$ satisfying (C1), there is a unique $\sigma' : \mathbb{N} \rightarrow D_\phi^{\mathrm{sat}}$ satisfying (C3) and (C4).*

(II) *Given $\sigma' : \mathbb{N} \rightarrow D_\phi^{\mathrm{sat}}$ satisfying (C3), there is a map $\sigma : \mathbb{N} \times (\{x_1, \ldots, x_s\} \times \{0, \ldots, l-1\}) \rightarrow \mathbb{Z}$ satisfying (C1) and (C4).*

Lemma 7(I) is easily shown by using the equality in (C4) to construct σ'. Observe that in Lemma 7(II), σ is not necessarily unique. The proof of Lemma 7(II) uses the existence of a map $h : D_\phi^{\mathrm{sat}} \rightarrow \mathbb{Z}^k$ such that for every $u \in D_\phi^{\mathrm{sat}}$, $[h(u)]^{\{1,\ldots,k\}} = u$ since D_ϕ^{sat} is the image of $[\cdot]^{\{1,\ldots,k\}}$ (Axiom of Choice).

As a corollary:

Lemma 8. *ϕ is satisfiable iff there is a structure $\sigma' : \mathbb{N} \to D_\phi^{\text{sat}}$ satisfying (C3) such that $\sigma', 0 \models \phi'$ where ϕ' is obtained from ϕ by replacing every occurrence of $\mathsf{X}^\beta x_\alpha$ by $x_{f(\langle x_\alpha, \beta \rangle)}$.*

In Lemma 8 above, we assume that $\sigma', i \models p[x_1 \leftarrow x_{f(\langle x_{j_1}, \beta_1 \rangle)}, \ldots, x_d \leftarrow x_{f(\langle x_{j_d}, \beta_d \rangle)}]$ holds true with $p \in \text{IPC}^{++}$ and p has free variables $x_1, \ldots, x_d$ whenever $p \overbrace{[x_1 \leftarrow x_{f(\langle x_{j_1}, \beta_1 \rangle)}, \ldots, x_d \leftarrow x_{f(\langle x_{j_d}, \beta_d \rangle)}]}^{\text{renaming}} \wedge \text{IPC}^{++}(\sigma'(i))$ is IPC^{++} satisfiable where $\text{IPC}^{++}(.)$ is the map defined in Sect. 3.1. For the Boolean and temporal operators, the satisfiability relation $\models$ on structures σ' is defined in the obvious homomorphic way. The abstraction of PLTL^{mod} models is now quite satisfying since the domain of σ' in Lemma 8 is finite and is of exponential cardinality in $|\phi|$.

4.2 Büchi automata

Using the approach for LTL reducing model checking and satisfiability problems to the emptiness problem for Büchi automata [VW94], we construct a Büchi automaton $\mathcal{A}_\phi$ on the alphabet D_ϕ such that $\mathrm{L}(\mathcal{A}_\phi)$, the language recognized to $\mathcal{A}_\phi$, is non-empty iff ϕ is PLTL^{mod} satisfiable. The automaton $\mathcal{A}_\phi$ is defined as the intersection of the following automata.

1. The Büchi automaton $\mathcal{A}_{D_\phi^{\text{sat}}}$ recognizes all the ω-sequences in $(D_\phi^{\text{sat}})^\omega$. $\mathcal{A}_{D_\phi^{\text{sat}}}$ is defined as the structure $\langle Q, Q_0, \to, F \rangle$ such that $Q = Q_0 = F = D_\phi$ and $u \xrightarrow{u''} u'$ iff $u = u''$ and $u \in D_\phi^{\text{sat}}$. By Corollary 1, one can check in polynomial space in $|\phi|$ whether $u \xrightarrow{u''} u'$.
2. The Büchi automaton $\mathcal{A}_{(C3)}$ recognizes the ω-sequences satisfying (C3). $\mathcal{A}_{(C3)}$ is defined as the structure $\langle Q, Q_0, \to, F \rangle$ such that $Q = Q_0 = F = D_\phi$ and $u \xrightarrow{u''} u'$ iff $u = u''$ and if $u = \langle \langle t_1, \ldots, t_k \rangle, X \rangle$ and $u' = \langle \langle t'_1, \ldots, t'_k \rangle, X' \rangle$ then $\langle t_{s+1}, \ldots, t_k \rangle = \langle t'_1, \ldots, t'_{k-s} \rangle$ and $X \cap \{s+1, \ldots, k\}^2 = \{\langle u+s, v+s \rangle : \langle u, v \rangle \in X',\ u+s \leq k,\ v+s \leq k\}$. One can check in polynomial time in $|\phi|$ whether $u \xrightarrow{u''} u'$.
3. The Büchi automaton $\mathcal{A}_{\text{PLTL}}$ recognizes the ω-sequences in D_ϕ^{sat} that satisfying ϕ (with the extended version of the relation $\models$).

The rest of this section is dedicated to construct $\mathcal{A}_{\text{PLTL}}$ based on developments from [LMS02] and on the abstraction introduced in Sect. 4.1. As usual, we define $cl(\phi)$, the closure of ϕ, as the smallest set of formulae such that

1. $\{\phi, \mathsf{X}^{-1}\top, \top\} \subseteq cl(\phi)$ and $cl(\phi)$ is closed under subformulae;
2. $cl(\phi)$ is closed under negation (we identify $\neg\neg\psi$ with ψ);
3. $\psi \mathsf{U} \psi' \in cl(\phi)$ implies $\mathsf{X}(\psi \mathsf{U} \psi') \in cl(\phi)$;
4. $\psi \mathsf{S} \psi' \in cl(\phi)$ implies $\mathsf{X}^{-1}(\psi \mathsf{S} \psi') \in cl(\phi)$.

The cardinality of $cl(\phi)$ is polynomial in $|\phi|$. We define an *atom* of ϕ to be a maximally consistent subset of $cl(\phi)$ defined as follows. X is an atom of ϕ iff

- $X \subseteq cl(\phi)$ and $\top \in X$;
- for every $\psi \in cl(\phi)$, $\psi \in X$ iff not $\neg\psi \in X$;
- for every $\psi \wedge \psi' \in cl(\phi)$, $\psi \wedge \psi' \in X$ iff $\psi \in X$ and $\psi' \in X$;
- for every $\psi \mathsf{U} \psi' \in cl(\phi)$, $\psi \mathsf{U} \psi' \in X$ iff either $\psi' \in X$ or $\{\psi, \mathsf{X}(\psi \mathsf{U} \psi')\} \subseteq X$;
- for every $\psi \mathsf{S} \psi' \in cl(\phi)$, $\psi \mathsf{S} \psi' \in X$ iff either $\psi' \in X$ or $\{\psi, \mathsf{X}^{-1}(\psi \mathsf{S} \psi')\} \subseteq X$;
- for every $\mathsf{X}^{-1}\psi \in cl(\phi)$, $\mathsf{X}^{-1}\psi \in X$ implies $\mathsf{X}^{-1}\top \in X$.

We can now define the generalized Büchi automaton $\mathcal{A}_{\text{PLTL}} = (Q, Q_0, \longrightarrow, \mathcal{F})$ with $\mathcal{F} = \{F_1, \ldots, F_m\} \subseteq \mathcal{P}(Q)$. A run $\rho : \mathbb{N} \to Q$ is accepting according to $\mathcal{F}$ iff for each $i \in \{1, \ldots, m\}$, $\rho(j) \in F_i$ for infinitely many $j \in \mathbb{N}$. A generalized Büchi condition can be easily converted to a Büchi condition by augmenting the states with a 0–m counter. The elements of $\mathcal{A}_{\text{PLTL}}$ are defined as follows:

- $Q = \mathcal{P}(cl(\phi))$; $Q_0 = \{X \in Q : \{\phi, \neg\mathsf{X}^{-1}\top\} \subseteq X\}$.
- $X \xrightarrow{u} Y$ iff
  (ATOM) X and Y are atoms of ϕ.
  (IPC^{++}) for every atomic p in X, $p' \wedge \text{IPC}^{++}(u)$ is IPC^{++}-satisfiable where p' is obtained from p by replacing the occurrences of $\mathsf{X}^{\beta} x_{\alpha}$ by $x_{f(\langle x_{\alpha}, \beta \rangle)}$.
  (NEXT) for each $\mathsf{X}\psi \in cl(\phi)$, $\mathsf{X}\psi \in X$ iff $\psi \in Y$.
  (PREVIOUS) for each $\mathsf{X}^{-1}\psi \in cl(\phi)$, $\mathsf{X}^{-1}\psi \in Y$ iff $\psi \in X$.
- Let $\{\psi_1 \mathsf{U} \varphi_1, \ldots, \psi_m \mathsf{U} \varphi_m\}$ be the set of until formulas in $cl(\phi)$. $\mathcal{F} = \{F_1, \ldots, F_m\}$ with for every $i \in \{1, \ldots, m\}$, $F_i = \{Z \in Q \mid \psi_i \mathsf{U} \varphi_i \notin Z \text{ or } \varphi_i \in Z\}$.

In $\mathcal{A}_{\text{PLTL}}$, one can check whether $X \xrightarrow{u} Y$ holds true in polynomial space in $|\phi|$. The conditions (ATOM), (NEXT), and (PREVIOUS) can be checked in polynomial-time in $|\phi|$. However, the above condition (IPC^{++}) requires polynomial space by Corollary 1. The main difference with LTL with past remains in the condition at the atomic level, involving here an IPC^{++}-satisfiability check.

Theorem 4. *ϕ is satisfiable iff $\mathrm{L}(\mathcal{A}_{\phi})$ is non-empty.*

This is a consequence of Lemma 8 and of the construction of Büchi automata from formulae in LTL with past [LMS02].

Viewing a model of ϕ as an ω-sequence of elements from D_{ϕ}, every formula ϕ defines an ω-regular subset of D_{ϕ}^{ω}, which can be also viewed as an ω-regular set of constraints by using the map IPC$^{++}(u)$. By contrast, in LTL($\{x = y, x < y\}$), the extension of LTL where the atomic formulae are of the form $\mathsf{X}^{n_1} x_1 \sim \mathsf{X}^{n_2} x_2$ with $\sim \in \{<, =\}$, there exist formulae that defines non ω-regular sets of constraints [DD03].

4.3 Complexity bounds

It is now standard to prove Theorem 5 below.

Theorem 5. *Satisfiability for* PLTL$^{\text{mod}}$ *is in* PSPACE.

Proof. $\mathcal{A}_\phi$ is defined as $\mathcal{A}_{D_\phi^{\mathrm{sat}}} \cap \mathcal{A}_{(C3)} \cap \mathcal{A}_{\mathrm{PLTL}}$. By the above considerations even though $\mathcal{A}_\phi$ has an exponential amount of states in $|\phi|$, checking the emptiness of $\mathrm{L}(\mathcal{A}_\phi)$ can be done on the fly in polynomial space in $|\phi|$ with a non-deterministic algorithm. As usual, by Savitch's theorem, this provides the requires PSPACE upper bound.

The PSPACE-hardness of the satisfiability problem for $\mathrm{PLTL}^{\mathrm{mod}}$ is a mere consequence of the PSPACE-hardness of plain LTL [SC85]. Moreover, it is worth observing that all the temporal operators in $\mathrm{PLTL}^{\mathrm{mod}}$ are MSO-definable and by using [GK03], it is not difficult to show that any extension of $\mathrm{PLTL}^{\mathrm{mod}}$ obtained by adding a finite amount of MSO-definable temporal operators remains in PSPACE.

This PSPACE upper bound is quite remarkable: in [BC02,DD02,DD03] PSPACE-completeness has been mainly established for extensions of LTL over concrete domains with satisfiability problem in P(only).

5 Adding logical first-order quantifiers

In this section, we investigate the extension of $\mathrm{PLTL}(\mathrm{IPC}^+)$ with the existential quantifier $\exists$, i.e. temporal operators can be in the scope of the existential quantifier $\exists$. This extension is denoted by $\mathrm{PLTL}^{\exists}(\mathrm{IPC}^+)$. In the general case, first-order LTL is known to be highly undecidable [Aba89,Krö90] even in the case the uninterpreted domains are finite [Tra63]. Similarly, first-order LTL over finite time structures is also highly undecidable [CMP99]. The decidability/complexity results obtained in this section are due to the fact that we can interpret any first-order formula ϕ of $\mathrm{PLTL}^{\exists}(\mathrm{IPC}^+)$ in a fixed concrete (infinite) domain that can be abstracted by a finite domain whose cardinality is nevertheless exponential in $|\phi|$. A similar argument cannot be used for $\mathrm{PLTL}^{\mathrm{mod}}$ augmented with the quantifier $\exists$ (denoted by $\mathrm{PLTL}^{\exists}(\mathrm{IPC}^{++})$) and the decidability status of this extension is unknown, see e.g. Table 1 in Sect. 7 for further details.

In order to define $\mathrm{PLTL}^{\exists}(\mathrm{IPC}^{++})$, the clause below is added to the definition of $\models$:

$$\sigma, i \models \exists\ y\ \phi \ \stackrel{\mathrm{def}}{\Leftrightarrow}\ \text{there exists } z \in \mathbb{Z} \text{ such that } \sigma', i \models \phi,$$

where σ' is defined as follows:

- for all $j \in \mathbb{N}$ and $x \in \mathrm{VAR} \setminus \{y\}$, $\sigma'(j, x) = \sigma(j, x)$;
- for every $j \in \mathbb{N}$, $\sigma'(j, y) = z$.

In σ' the interpretation of the variable y is rigid unlike other possible variables. For the sake of clarity, we shall only consider $\mathrm{PLTL}^{\exists}(\mathrm{IPC}^{++})$ formulae ϕ in which if $\exists y\ \psi$ occurs as a subformula of ϕ, the only occurrences of the variable y in ϕ are in ψ. The variable y will be said to be a global variable (by reference to its rigidity). By contrast, the free variables of ϕ will be said to be local. In that sense, all the variables occurring in $\mathrm{PLTL}^{\mathrm{mod}}$ formulae are local.

We shall also consider an interesting fragment of $\mathrm{PLTL}^{\exists}(\mathrm{IPC}^{++})$ in which the use of $\exists$ at the logical level is restricted. We write $\mathrm{PLTL}^{\downarrow}(\mathrm{IPC}^{++})$ to denote the fragment of $\mathrm{PLTL}^{\downarrow}(\mathrm{IPC}^{++})$ where the quantifier $\exists$ is used only in formulae of the form:

$$\exists x' \ (x' = \mathsf{X}^i x) \wedge \phi,$$

with $i \geq 0$. Observe that $\exists x' \ x' = \mathsf{X}^i x \wedge \phi$ is not equivalent to $\phi[x' \leftarrow \mathsf{X}^i x]$, the formula obtained from ϕ where each occurrence of x' is replaced by $\mathsf{X}^i x$. Indeed, replacement of x' may be done in the scope of temporal operators which is incompatible with the interpretation of $\mathsf{X}^i x$ at the current state.

In the sequel, we write $\downarrow_{x'=\mathsf{X}^i x} \phi$ instead of $\exists x' \ x' = \mathsf{X}^i x \wedge \phi$. The freeze quantifier $\downarrow$ that allows to bind the values of variables to a fixed value is a powerful binder used for instance in real-time logics (see e.g., [AH93,AH94]) and in hybrid logics (see e.g., [Gor96,Bla00]). Adding this kind of operator can easily lead to undecidability (see e.g., [Gor96]) when no restriction is required on the Kripke structures. Observe that the decidability status of decidable LTL over concrete domains from [DD03] augmented with the freeze operator is still open. In this section, we treat a very particular case with integer periodicity constraints for which decidability follows from decidability of $\mathrm{PLTL}^{\mathrm{mod}}$.

5.1 EXPSPACE lower bound

Adding the freeze operator $\downarrow$ to $\mathrm{PLTL}(\mathrm{IPC}^{++})$ leads to an exponential blow-up even if only future-time operators and simple periodicity constraints are used.

Theorem 6. *Satisfiability for $\mathrm{PLTL}^{\downarrow}(\mathrm{IPC}^{++})$ restricted to future-time operators and simple periodicity constraints is* EXPSPACE*-hard.*

As a corollary, satisfiability for $\mathrm{PLTL}^{\exists}(\mathrm{IPC}^{++})$ is also EXPSPACE-hard.

Proof. We prove this result by a reduction from the 2^n-corridor tiling problem that is EXPSPACE-complete, see [vEB97] and references therein. A tile is a unit square of one of the several tile-types and the tiling problem we considered is specified by means of a finite set T of tile-type (say $T = \{t_1, \ldots, t_k\}$), two binary relations H and V over T and two distinguished tile-types t_{init}, $t_{final} \in T$. The tiling problem consists in determining whether, for a given number n in unary, the region $[0, \ldots, 2^n - 1] \times [0, \ldots, k-1]$ of the integer plane for some k can be tiled consistently with H and V, t_{init} is the left bottom tile, and t_{final} is the right upper tile.

Given an instance $I = \langle T, t_{init}, t_{final}, n\rangle$ of the tiling problem, we build a formula ϕ_I such that $I = \langle T, t_{init}, t_{final}, n\rangle$ has a solution iff ϕ_I is $\mathrm{PLTL}^{\downarrow}(\mathrm{IPC}^{++})$ satisfiable. We introduce the variables in $\mathrm{PLTL}^{\downarrow}(\mathrm{IPC}^{++})$ that are needed in the sequel:

- *pos* is a variable that allow to count until 2^n; there is a corresponding global variable *pos*$'$; each element $\langle \alpha, i\rangle$ of a row $[0, \ldots, 2^n - 1] \times \{i\}$ satisfies $pos \equiv_{2^n}$

α; the finite region $[0,\ldots,2^n-1]\times[0,\ldots,k-1]$ will be encoded by the following prefix of a $\mathrm{PLTL}^{\downarrow,\mathrm{mod}}$ model

$$(\{pos \equiv_{2^n} 0\}\cdot\{pos \equiv_{2^n} 1\}\cdot\ldots\cdot\{pos \equiv_{2^n} 2^n-1\})^k;$$

- for $t\in T$, z_t is a variable such that $D_t := z_t \equiv_2 0$ is the formula encoding the fact that at a certain position of the integer plane the tile t is present. There is also a global variable z'_t and $D'_t := (z'_t \equiv_2 0)$;
- end and we define $\mathtt{END} := (end \equiv_2 0)$.

The formula ϕ_I is the conjunction of the following formulae:

- $\neg\mathtt{END}\wedge(\neg\mathtt{END}\mathsf{U}(pos\equiv_{2^n} 0\wedge\Box\mathtt{END}))$ (the region of the integer plane for the solution is finite);
- $\Box(\neg\mathtt{END}\Rightarrow\bigvee_{t\in T}(D_t\wedge\bigwedge_{t'\neq t}\neg D_{t'}))$; (exactly one tile per element of the plane region);
- $\Diamond(pos\equiv_{2^n}(2^n-1)\wedge\neg\mathtt{END}\wedge D_{t_{final}}\wedge\mathsf{X}\mathtt{END})$ (right upper tile);
- $pos\equiv_{2^n} 0\wedge D_{t_{init}}$ (left bottom tile);
- $\Box(\mathsf{X}pos\equiv_{2^n} pos+1)$ (incrementation of the counter pos modulo 2^n);
- $\Box((\overbrace{(\neg pos\equiv_{2^n}(2^n-1))}^{\text{not the last element of a row}}\wedge\neg\mathtt{END})\Rightarrow\bigwedge_{t\in T}(D_t\Rightarrow\bigvee_{\langle t,t_{hor}\rangle\in H}\mathsf{X}D_{t_{hor}}))$ (horizontal consistency);
-
$$\Box(\neg\mathtt{END}\wedge\overbrace{\Diamond(\mathsf{X}\neg\mathtt{END}\wedge pos\equiv_{2^n}(2^n-1))}^{\text{not on the last row}}\Rightarrow\downarrow_{pos'=pos}\downarrow_{z'_{t_1}=z_{t_1}}\cdots\downarrow_{z'_{t_k}=z_{t_k}}$$
$$\mathsf{X}((\neg(pos'\equiv_{2^n} pos))\mathsf{U}(pos'\equiv_{2^n} pos\wedge\bigwedge_{t\in T}(D'_t\Rightarrow\bigvee_{\langle t,t_{ver}\rangle\in V}D_{t_{ver}})))$$
(vertical consistency).
The last part of the above formula allows us to go exactly to the cell above a given cell and check the vertical consistency. Observe that $\downarrow$ is present in ϕ_I only to express the vertical consistency.

One can show that the instance $I=\langle T,t_{init},t_{final},n\rangle$ has a solution iff ϕ_I is $\mathrm{PLTL}^{\downarrow}(\mathrm{IPC}^{++})$ satisfiable.

In the EXPSPACE-hardness proof, we are able to count till 2^n using only a number of resources polynomial in n (this can be done also in LTL) and we can compare the truth value of atomic formulae in states of "temporal distance" exactly 2^n using only a polynomial amount of symbols (this cannot be done in $\mathrm{PLTL}^{\mathrm{mod}}$), whence the reduction of a famous EXPSPACE-complete tiling problem. In the proof, the freeze operator is used to characterize a distance equal to 2^n exactly with a polynomial amount of symbols.

5.2 EXPSPACE upper bound

An exponential-time translation from $\mathrm{PLTL}^{\exists}(\mathrm{IPC}^{+})$ to $\mathrm{PLTL}(\mathrm{IPC}^{+})$ allows us to show the following result.

Lemma 9. *Satisfiability for* $\mathit{PLTL}^{\exists}(\mathrm{IPC}^{+})$ *is in* EXPSPACE.

Proof. Let ϕ be a formula of $\mathrm{PLTL}^{\exists,\mathrm{mod}}$ with

- free variables $x_1, \ldots, x_k$;
- $c_1 < \ldots < c_n$ are the constants in ϕ occurring in constraints of the form $x \sim c$ with $\sim \in \{<, >, =\}$;
- natural numbers $k_1, \ldots, k_u$ occurring in the context of $\equiv$-atomic formulae and their lcm is denoted by K.

Let D be the set

$$\{\langle i, j, l\rangle \in \{0, \ldots, n+1\}^2 \times \{0, \ldots, K-1\} : \mathrm{IPC}^{<}((\langle i, j, l\rangle, 1)) \text{ is satisfiable}\}.$$

To each $\langle i, j, l\rangle \in D$, we associate a constant $d_{\langle i,j,l\rangle}$ such that $|d_{\langle i,j,l\rangle}|$ is polynomial in $|\phi|$ and $[x_1 \leftarrow d_{\langle i,j,l\rangle}] \models \mathrm{IPC}^{<}((\langle i, j, l\rangle, 1))$.

We reduce $\mathrm{PLTL}^{\exists}(\mathrm{IPC}^{+})$ satisfiability to $\mathrm{PLTL}(\mathrm{IPC}^{+})$ satisfiability. The translation t is the following:

- $t(p) = p$ for p atomic,
- t is homomorphic for the Boolean and temporal operators,
- $t(\exists\ x_\alpha\ \psi) = \bigvee_{\langle i,j,l\rangle \in D} t(\psi[x_\alpha \Leftarrow d_{\langle i,j,l\rangle}])$ where $\psi[x_\alpha \Leftarrow d_{\langle i,j,l\rangle}]$ denotes the formula obtained from ψ by replacing occurrences of x_α by $d_{\langle i,j,l\rangle}$ with adequate simplications. This is an homomorphic operation on the Boolean and temporal operators. By way of example,

  $$(x \equiv_k y + [c_1, c_2])[y \Leftarrow d]$$

  with $d = s \times k + c$, $s \in \mathbb{Z}$, and $c \in \{0, \ldots, k-1\}$ is equal to

  $$\bigvee_{c_1 \leq \alpha \leq c_2} x \equiv_k (c + \alpha).$$

  Similarly, $(x \sim c)[x \Leftarrow d]$ with $\sim \in \{=, <, >\}$ takes either the value $\top$ or $\perp$ depending whether $d \sim c$ holds true.

For instance $(x \equiv_3 x_\alpha + [1, 2])[x_\alpha \Leftarrow 5]$ is equal to $x \equiv_3 0 \vee x \equiv_3 1$. One can check that $|t(\phi)|$ is in $2^{\mathcal{O}(|\phi|^2)}$. Even if we add atomic constraints of the form $x \equiv_k [c_1, c_2]$ in the definition of IPC^{+} (with the obvious interpretation), one cannot avoid the quadratic exponent in $2^{\mathcal{O}(|\phi|^2)}$.

It is then easy to show that ϕ is $\mathrm{PLTL}^{\exists,\mathrm{mod}}$ satisfiable iff $t(\phi)$ is $\mathrm{PLTL}(\mathrm{IPC}^{+})$ satisfiable.

The above reduction does not work if we allow atomic constraints of the form $x = y$ (belonging to IPC^{++}) as in the formula $\Box \downarrow_{x'=x} \mathsf{X}\Box(x \neq x')$ that characterizes models where all the values for x are different. Such a formula is particularly interesting since in cryptographic protocols, nonces, ideally variables that never take twice the same value, are often used to guarantee freshness properties. Hence, this can be specified in PLTL$^{\mathrm{mod}}$ with $\downarrow$. So far, the decidability status of PLTL$^{\mathrm{mod}}$ augmented with the freeze operator $\downarrow$ is an open problem as well as the EXPSPACE-hardness of PLTL$^{\exists}$(IPC^{+}).

6 Application to the equivalence problem for extended single-string automata

In this section, we characterize the complexity of the equivalence problem for extended single-string automata defined in [LM01, Sect. 5]. This problem is central to check whether two time granularities are equivalent (see also [Wij00]) when granularities are encoded by such automata. Such automata can be viewed as Büchi automata recognizing exactly one ω-word. Guards on transitions expressed by integer periodicity constraints and update maps on transitions provide conciseness of such constraint automata. Unlike timed automata, no synchronization between variables is performed and the languages for guards and update maps are quite different, see e.g. [AD94,BDFP00]. We improve the known EXPSPACE upper bound from [LM01] into a PSPACE upper bound by reducing the equivalence problem to the model-checking problem for PLTL$^{\mathrm{mod}}$-automata. Moreover, although a seemingly efficient algorithm is presented in [LM01], we show the PSPACE-hardness by reducing QBF.
Let IPC* be the fragment of IPC$^{\{\exists\}}$ containing Boolean combinations of atomic constraints of the form either $x \equiv_k c$ or $\exists z\ (x \equiv_k z \ \wedge\ y \equiv_{k'} z)$. Elements of IPC* will be guards on transitions. An update map g for the variable x_i is of the form either $x_i := x_i + c$ or $x_i := c$ with $c \in \mathbb{Z}$. We write $\mathrm{UP}_{x_1,\ldots,x_n}$ to denote the set of update maps for some set $\{x_1, \ldots, x_n\}$ of variables.

An extended single-string automaton $\mathcal{A}$ (ESSA) over the finite set of variables $\{x_1, \ldots, x_n\}$ [LM01] is a structure of the form $\langle Q, q_0, \overline{v_0}, \Sigma, \delta\rangle$ where

- Q is a finite set of states and $q_0 \in Q$ (initial state);
- $\overline{v_0} \in \mathbb{Z}^n$ (initial value of the variables $x_1, \ldots, x_n$);
- Σ is a finite alphabet;
- $\delta \subseteq Q \times \Sigma \times Q \times (\{\top\} \cup \mathrm{IPC}^{*}) \times \mathcal{P}(\mathrm{UP}_{x_1,\ldots,x_n})$ and for every $q \in Q$,
    1. either there is a unique u such that $\langle q, u\rangle \in \delta$, u is of the form $\langle a, q', \top, X\rangle$, and X contains exactly one update map per variable x_i;
    2. or there are exactly two u such that $\langle q, u\rangle \in \delta$, say u_1 and u_2, and in that case u_1 is of the form $\langle a_1, q_1, p, X_1\rangle$, u_2 is of the form $\langle a_2, q_2, \neg p, X_2\rangle$ where p is a constraint in IPC* built over variables in $\{x_1, \ldots, x_n\}$ and in both X_1 and X_2 exactly one update map for x_i is present.

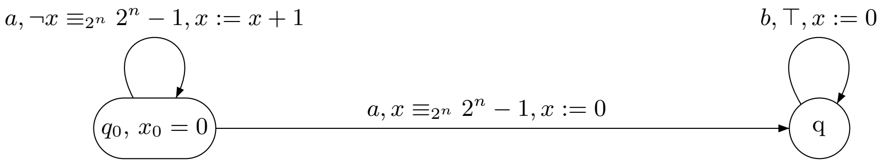
Case 1. is subsumed by Case 2. by taking $p = \top$. The elements of δ are also denoted by $q \xrightarrow{a,p,X} q'$ (p is the guard and X is the global update map).

A configuration is a member $\langle q, \overline{v} \rangle \in Q \times \mathbb{Z}^n$. We define the one-step relation $\xrightarrow{a}$ for $a \in \Sigma$ as follows: $\langle q, \overline{v} \rangle \xrightarrow{a} \langle q', \overline{v'} \rangle$ iff there is $\langle q, a, q', X, p \rangle \in \delta$ such that $[x_1 \leftarrow v_1, \ldots, x_n \leftarrow v_n] \models p$ (in IPC^{++}) and for every $g \in X$,

- if g is $x_i := x_i + c$ then $v'_i = v_i + c$;
- if g is $x_i := c$ then $v'_i = c$.

It is easy to check that there is exactly one sequence $w = a_1 a_2 \ldots \in \Sigma^\omega$ such that $\langle q_0, \overline{v_0} \rangle \xrightarrow{a_1} \langle q_1, \overline{v_1} \rangle \xrightarrow{a_2} \ldots$. The unique ω-sequence generated from the ESSA $\mathcal{A}$ is denoted by $w_{\mathcal{A}}$. The equivalence problem for ESSA consists in checking whether $w_{\mathcal{A}} = w_{\mathcal{A}'}$, given two ESSA $\mathcal{A}$ and $\mathcal{A}'$. This problem introduced in [LM01] is central to check the equivalence of time granularities when granularities are encoded by such automata. Condition 2. in the definition of the transition relation δ has been introduced in [LM01] (in a slightly different form but equivalent to ours) in order to handle priorities between transitions.

For instance, the ω-word associated with the ESSA below is $a^{2^n} \cdot b^\omega$:



Lemma 10. *The equivalence problem for ESSA can be solved in* PSPACE.

Proof. Given two ESSA $\mathcal{A}$ and $\mathcal{A}'$, one can build an $\mathrm{LTL}(\mathrm{IPC}^{\{\exists\}})$-automaton $\mathcal{B}$ in polynomial time such that $l(\mathcal{B})$ is non-empty (equivalent to $\mathcal{B} \models_\exists \top$) iff $w_{\mathcal{A}} = w_{\mathcal{A}'}$. Actually $l(\mathcal{B})$ will contain at most one ω-word. The $\mathrm{LTL}(\mathrm{IPC}^{\{\exists\}})$-automaton $\mathcal{B}$ is indeed a kind of product of $\mathcal{A}$ and $\mathcal{A}'$.

Let $\mathcal{A} = \langle Q, q_0, \overline{v_0}, \Sigma, \delta \rangle$ and $\mathcal{A}' = \langle Q', q'_0, \overline{v'_0}, \Sigma, \delta' \rangle$ be ESSA over the (disjoint) sets of variables $\{x_1, \ldots, x_{n_1}\}$ and $\{x_{n_1+1}, \ldots, x_{n_1+n_2}\}$, respectively. We build an $\mathrm{LTL}(\mathrm{IPC}^{\{\exists\}})$-automaton $\mathcal{B}$ such that $l(\mathcal{B})$ is non-empty (equivalent to $\mathcal{B} \models_\exists \top$) iff $w_{\mathcal{A}} = w_{\mathcal{A}'}$ (actually $l(\mathcal{B})$ will contain at most one ω-word). We write K to denote the lcm of all the constants k occurring in one of the two input automata in the context of $\equiv_k$. The constructed automaton $\mathcal{B}$ has a quite restricted form since the labels on transitions are atomic formulae from $\mathrm{LTL}(\mathrm{IPC}^{\{\exists\}})$.

With each update map g occurring in one of the two input automata, we associate an atomic formula $\mathrm{PLTL}^{\mathrm{mod}}(g)$ in $\mathrm{PLTL}^{\mathrm{mod}}$ as follows:

- if g is of the form $x_i := x_i + c$, then $\mathrm{PLTL}^{\mathrm{mod}}(g)$ is equal to $\mathsf{X}x_i \equiv_K x_i + c_K$ where c_K is the unique element of $\{0, \ldots, K-1\}$ such that $c = c_K + \alpha \times K$ for some $\alpha \in \mathbb{Z}$;
- if g is of the form $x_i := c$, then $\mathrm{PLTL}^{\mathrm{mod}}(g)$ is equal to $\mathsf{X}x_i \equiv_K c_K$.

Let $\mathcal{B} = \langle Q'', init, F, \delta'' \rangle$ be the $\mathrm{LTL}(\mathrm{IPC}^{\{\exists\}})$-automaton defined as follows:

- $Q'' = Q \times Q' \cup \{init\}$ and $F = Q''$;

- $init \xrightarrow{\phi} \langle q_1, q_1' \rangle$ iff the following conditions are verified:
  - $\langle q_0, a_1, q_1, p_1, X_1 \rangle, \langle q_0', a_2, q_1', p_2, X_2 \rangle \in \delta$ with $a_1 = a_2$ (the same letter is read);
  - ϕ is the conjunction of the following formulae:
    1. $\bigwedge_{i=1}^{n_1} x_i \equiv_K c_i$ where for each i, $c_i \in \{0, \ldots, K-1\}$ and $v_{0,i} \equiv_K c_i$ (initial condition of $\mathcal{A}$);
    2. $\bigwedge_{i=n_1+1}^{n_2} x_i \equiv_K c_i$ where for each i, $c_i \in \{0, \ldots, K-1\}$ and $v'_{0,i-n_1} \equiv_k c_i$ (initial condition of $\mathcal{A}'$);
    3. $\bigwedge_{g \in X_1 \cup X_2} \mathrm{PLTL}^{\mathrm{mod}}(g)$ (synchronization of $\mathcal{A}$ and $\mathcal{A}'$ for update maps);
    4. $p_1 \wedge p_2$ (synchronization of $\mathcal{A}$ and $\mathcal{A}'$ for guards).
- $\langle q_1, q_1' \rangle \xrightarrow{\phi} \langle q_2, q_2' \rangle$ iff the following conditions are verified:
  - $\langle q_1, a_1, q_2, p_1, X_1 \rangle, \langle q_1', a_2, q_2', p_2, X_2 \rangle \in \delta$ with $a_1 = a_2$;
  - ϕ is $\bigwedge_{g \in X_1 \cup X_2} \mathrm{PLTL}^{\mathrm{mod}}(g) \wedge p_1 \wedge p_2$.

It is easy to check that $\mathcal{B}$ can be built in polynomial-time in the size of $\mathcal{A}$ and $\mathcal{A}'$. Moreover, one can show that $l(\mathcal{B})$ is non-empty iff $w_{\mathcal{A}} = w_{\mathcal{A}'}$.

However, it is worth observing that the transformation of the update map $x := x + c$ into the atomic $\mathrm{PLTL}^{\mathrm{mod}}$ formula $\mathsf{X}x \equiv_K x + c_K$ is correct because the guards in $\mathcal{A}$, $\mathcal{A}'$ are in IPC^*. Indeed, IPC^* cannot distinguish $n = m + c$ from $n \equiv_K m + c_K$.

The above result of PSPACE upper bound can be extended if in the definition of ESSA, the constraint language IPC^* is extended to $\mathrm{IPC}^{\{\exists,[]\}}$. However, adding constraints of the form $x = 0$ (that are not in $\mathrm{IPC}^{\{\exists,[]\}}$) would lead to undecidability by reduction from the halting problem for two-counter machines.

On the other side, one can show that the equivalence problem for ESSA is PSPACE-hard even if

- the constraints occurring in transitions are literals (conjunction and disjunction are disallowed) containing atomic constraints of the form $x \equiv_k c$;
- no update maps with incrementation or decrementation are used (it remains the update maps of the form either $x := x$ -identity- or $x := c$);
- the only k occurring in $\equiv_k$ is 2.
- the alphabet Σ is binary (if Σ is unary, then the equivalence problem is trivial). Observe that time granularities are encoded with only three symbols $\{\square, \blacksquare, \wr\}$ in [Wij00].

Lemma 11. *The equivalence problem for ESSA is* PSPACE*-hard.*

Proof. We reduce QBF to the equivalence problem for ESSA. Let ϕ be an instance of QBF of the form below:

$$\forall\, x_1\; \exists\, x_2\; \ldots \forall\, x_{2n-1}\; \exists\, x_{2n}\; \overbrace{\bigwedge_{i=1}^{m} (l_1^i \vee l_2^i \vee l_3^i)}^{\phi'}$$

where the l_j^is are literals over the propositional variables in $x_1, \ldots, x_{2n}$. In spite of the prenex form of ϕ, the strict alternation between $\forall$ and $\exists$, and the fact that ϕ' is in 3CNF, QBF restricted to such QBF formulae can be easily shown to be PSPACE-hard. We shall define, in logarithmic space in $|\phi|$, an ESSA $\mathcal{A}$ such that ϕ is satisfiable iff $w_{\mathcal{A}} = a^{\omega}$. This will lead to the PSPACE-hardness of the equivalence problem for ESSA since it is easy to design an ESSA $\mathcal{B}$ such that $w_{\mathcal{B}} = a^{\omega}$.
First, we recall in Figure 5 the standard recursive procedure EVAL(ψ, v) to solve QBF. The first argument is a QBF formula and v is an interpretation of propositional variables for a superset of the free variables in ψ.

EVAL(ψ, v)

- if $\psi = p$, then return $v(p)$;
- if $\psi = \psi_1 \wedge \psi_2$, then, if EVAL$(\psi_1, v) = 1$, then return EVAL$(\psi_2, v)$, otherwise return 0;
- if $\psi = \psi_1 \vee \psi_2$, then, if EVAL$(\psi_1, v) = 1$, then return 1, otherwise return EVAL(ψ_2, v);
- if $\psi = \neg\psi_1$, then return $1 -$ EVAL(ψ_1, v);
- if $\psi = \exists\ p\ \psi_1$, then if EVAL$(\psi_1, v[p \leftarrow 0]) = 1$, then return 1, otherwise return EVAL$(\psi_1, v[p \leftarrow 1])$;
- if $\psi = \forall\ p\ \psi_1$, then if EVAL$(\psi_1, v[p \leftarrow 0]) = 0$, then return 0, otherwise return EVAL$(\psi_1, v[p \leftarrow 1])$;

Fig. 5. Standard procedure to solve QBF in polynomial space

ϕ is QBF satisfiable iff EVAL$(\phi, [])$ return 1, see e.g. [HU79, Sect. 13.4] where [] denotes the empty interpretation. From the execution of EVAL$(\phi, [])$, we can extract a sequence of the form

$$\langle \mathrm{EVAL}(\phi', v_1), A_1 \rangle, \ldots, \langle \mathrm{EVAL}(\phi', v_\alpha), A_\alpha \rangle,$$

that corresponds to the successive calls to EVAL with the first argument being the matrix ϕ' and A_i is the value returns by EVAL(ϕ', v_i). Without any loss of generality, we can assume that each v_i belongs to $\{0,1\}^{2n}$. Observe that $v_1 = 0^{2n}$, $A_\alpha =$ EVAL$(\phi, [])$, and $v_1, \ldots, v_\alpha$ is a strictly increasing sequence of natural numbers encoded by $2n$ bits.

We build an ESSA $\mathcal{A}$ that simulates the above-mentioned sequence of calls of the form EVAL(ϕ', v). Let $\mathcal{A}$ be the following structure $\langle Q, q_0, \overline{v_0}, \Sigma, \delta \rangle$;

- Q is the union of the following elements:
  - $\{\mathrm{LIT}_1^i, \mathrm{LIT}_2^i, \mathrm{LIT}_3^i : 1 \leq i \leq m\}$;
  - $\{\mathrm{rdepth}_i : 1 \leq i \leq 2n\}$ (rdepth_i is reached when we go to recursion depth i);

- $\{0,1\}$ (i stands for "EVAL(ϕ, []) returns i").

– $q_0 = \mathrm{LIT}_1^1$ and $\overline{v_0} = 0^{2n}$;

– $\Sigma = \{a, b\}$;

– Before defining the transition table δ we need some preliminary definitions. For every $j \in \{1, \ldots, 2n+1\}$ we write UPD_j to denote the following set of update maps:

$$\{x_k := x_k : 1 \leq k < j\} \cup \{x_j := 1\} \cup \{x_k := 0 : j < k \leq 2n\}.$$

The set UPD_{2n+1} does not modify the values of the variables in $\{x_1, \ldots, x_{2n}\}$ and therefore we denote it by ID. We define the map L that translates naturally literals into atomic constraints in the following way: $L(x_i) = (x_i \equiv_2 1)$ and $L(\neg x_i) = (x_i \equiv_2 0)$.
The encoding of the structure of ϕ' is done via the following transitions:

- For $j = 1, \ldots, m$, $\mathrm{LIT}_1^j \xrightarrow{a, L(l_1^j), \mathrm{ID}} \mathrm{LIT}_1^{j+1}$ (satisfaction of the clause $C_j = l_1^j \vee l_2^j \vee l_3^j$ thanks to the interpretation of l_1^j with the current values of the x_is) and $\mathrm{LIT}_1^j \xrightarrow{a, \neg L(l_1^j), \mathrm{ID}} \mathrm{LIT}_2^j$ (otherwise, check whether C_j is satisfied thanks to l_2^j). In the case $j = m$, the former transition is replaced by $\mathrm{LIT}_1^m \xrightarrow{a, L(l_1^m), \mathrm{ID}} \mathrm{rdepth}_{2n-1}$ (no more clauses need to be satisfied, start the process for reducing the recursion depth).
- For $j = 1, \ldots, m$, $\mathrm{LIT}_2^j \xrightarrow{a, L(l_2^j), \mathrm{ID}} \mathrm{LIT}_1^{j+1}$ and $\mathrm{LIT}_2^j \xrightarrow{a, \neg L(l_2^j), \mathrm{ID}} \mathrm{LIT}_3^j$. In the case $j = m$, the former transition is replaced by $\mathrm{LIT}_2^m \xrightarrow{a, L(l_2^m), \mathrm{ID}} \mathrm{rdepth}_{2n-1}$.
- For $j = 1, \ldots, m$, $\mathrm{LIT}_3^j \xrightarrow{a, L(l_3^j), \mathrm{ID}} \mathrm{LIT}_1^{j+1}$ (satisfaction of C_j thanks to the interpretation of l_3^j) and $\mathrm{LIT}_3^j \xrightarrow{a, \neg L(l_3^j), \mathrm{ID}} \mathrm{rdepth}_{2n}$ (the clause C_j is not satisfied by the current interpretation, start the process for reducing the recursion depth). In the case $j = m$, the former transition is replaced by $\mathrm{LIT}_3^m \xrightarrow{a, L(l_3^m), \mathrm{ID}} \mathrm{rdepth}_{2n-1}$.

The following transitions allow to branch to the appropriate recursion depth.

- for $j \in \{3, \ldots, 2n\}$, $\mathrm{rdepth}_j \xrightarrow{a, x_j \equiv_2 1, \mathrm{ID}} \mathrm{rdepth}_{j-2}$ and $\mathrm{rdepth}_j \xrightarrow{a, x_j \equiv_2 0, \mathrm{UPD}_j} \mathrm{LIT}_1^1$. Moreover, for $j \in \{1, 2\}$, we consider the following transitions: $\mathrm{rdepth}_1 \xrightarrow{a, x_1 \equiv_2 0, \mathrm{UPD}_1} \mathrm{LIT}_1^1$, $\mathrm{rdepth}_1 \xrightarrow{a, x_1 \equiv_2 1, \mathrm{ID}} 1$, $\mathrm{rdepth}_2 \xrightarrow{a, x_2 \equiv_2 0, \mathrm{UPD}_2} \mathrm{LIT}_1^1$, and $\mathrm{rdepth}_2 \xrightarrow{a, x_2 \equiv_2 1, \mathrm{ID}} 0$.
- $0 \xrightarrow{b, \mathrm{ID}, \top} 0$ and $1 \xrightarrow{a, \mathrm{ID}, \top} 1$.

It is easy to check that

$$\langle \mathrm{LIT}_1^1, v_1 \rangle \xrightarrow{a^+} \langle \mathrm{LIT}_1^1, v_2 \rangle \xrightarrow{a^+} \ldots \langle \mathrm{LIT}_1^1, v_\alpha \rangle$$

and within each path $\langle \mathrm{LIT}_1^1, v_i \rangle \xrightarrow{a^+} \langle \mathrm{LIT}_1^1, v_{i+1} \rangle$, there is no configuration of the form $\langle \mathrm{LIT}_1^1, v \rangle$. Moreover, for every $i \in \{1, \ldots, \alpha\}$, if $A_i = 1$, then $\langle \mathrm{LIT}_1^1, v_i \rangle \xrightarrow{a^+} \mathrm{rdepth}_{2n-1}$, otherwise $\langle \mathrm{LIT}_1^1, v_i \rangle \xrightarrow{a^+} \mathrm{rdepth}_{2n}$.

Moreover, if EVAL(ϕ, []) returns 1, then

$$\langle \mathrm{LIT}^1_1, v_\alpha \rangle \xrightarrow{a^+} \mathrm{rdepth}_{2n-1} \xrightarrow{a^+} 1 \xrightarrow{a} 1 \xrightarrow{a} 1 \ldots,$$

otherwise

$$\langle \mathrm{LIT}^1_1, v_\alpha \rangle \xrightarrow{a^+} \mathrm{rdepth}_{2n} \xrightarrow{a^+} 0 \xrightarrow{b} 0 \xrightarrow{b} 0 \ldots.$$

By way of example, we present in Figure 6, the ESSA for the QBF formula φ below:

$$\forall\ x_1\ \exists\ x_2\ \forall\ x_3\ \exists\ x_4\ (x_1 \vee x_2 \vee x_4) \wedge (\neg x_1 \vee x_3 \vee \neg x_4).$$

Observe that in the ESSA associated with φ, only the incoming transitions of LIT^1_1 modify the values of the variables. In order to simplify the figure, for some transition with letter a and set of update maps ID, we have only labelled the transition by the guard. By the way, it is easy to check that φ is satisfiable, by showing for instance that the run of the automaton explores the following interpretations of x_1, x_2, x_3, x_4:

$$0000,\ 0001,\ 0010,\ 0011,\ 1000,\ 1010.$$

Moreover, the ESSA associated with φ is not flat in the sense of [CJ98,CC00].

Theorem 7. *The equivalence problem for ESSA is* PSPACE*-complete.*

The proof of the PSPACE upper bound stated in Corollary 10 entails that checking whether $w_{\mathcal{A}} = w_{\mathcal{A}'}$ can be done in time

$$\mathcal{O}(2^{2 \times maxsize^2 \times n} \times |Q| \times |Q'|),$$

where n is the number of variables used in $\mathcal{A}, \mathcal{A}'$ and $maxsize$ is the size of the greatest integer k in $\equiv_k$-guards occurring in $\mathcal{A}, \mathcal{A}'$. Hence, the greatest integer occuring in $\mathcal{A}, \mathcal{A}'$ has value in $\mathcal{O}(2^{maxsize})$. Consequently, the parameterized version of the equivalence problem for ESSA is fixed-parameter tractable (FTP) when the parameters are the number of variables and the integers, see e.g. [DF99,DFS99] for definitions and motivations about the parameterized complexity paradigm. However, the proof of Lemma 11 entails that the problem remains PSPACE-hard when the only integer k in $\equiv_k$-guards occurring in $\mathcal{A}, \mathcal{A}'$ is 2 or when the integers are encoded with a unary representation. Similarly, the problem remains PSPACE-hard when only two distinct variables are used. Indeed, by following the construction of the proof of Lemma 11, the binary encoding of a first variable encodes a propositional valuation whereas the second variable is used as an auxiliary register to test the nullity of each bit of the first variable. Details are omitted here. By contrast, we are only able to prove the co-NP-hardness of the problem restricted to a unique variable. Hence, it is open whether the equivalence problem for ESSA restricted to a unique variable (but without restriction on the size of integers) is PSPACE-hard.

Another simpler problem which arises when dealing with time granularities, is to find the nth occurrence of a given symbol in a string [LMP03, Sect. 4]. Here is the definition of the occurrence problem for ESSA:

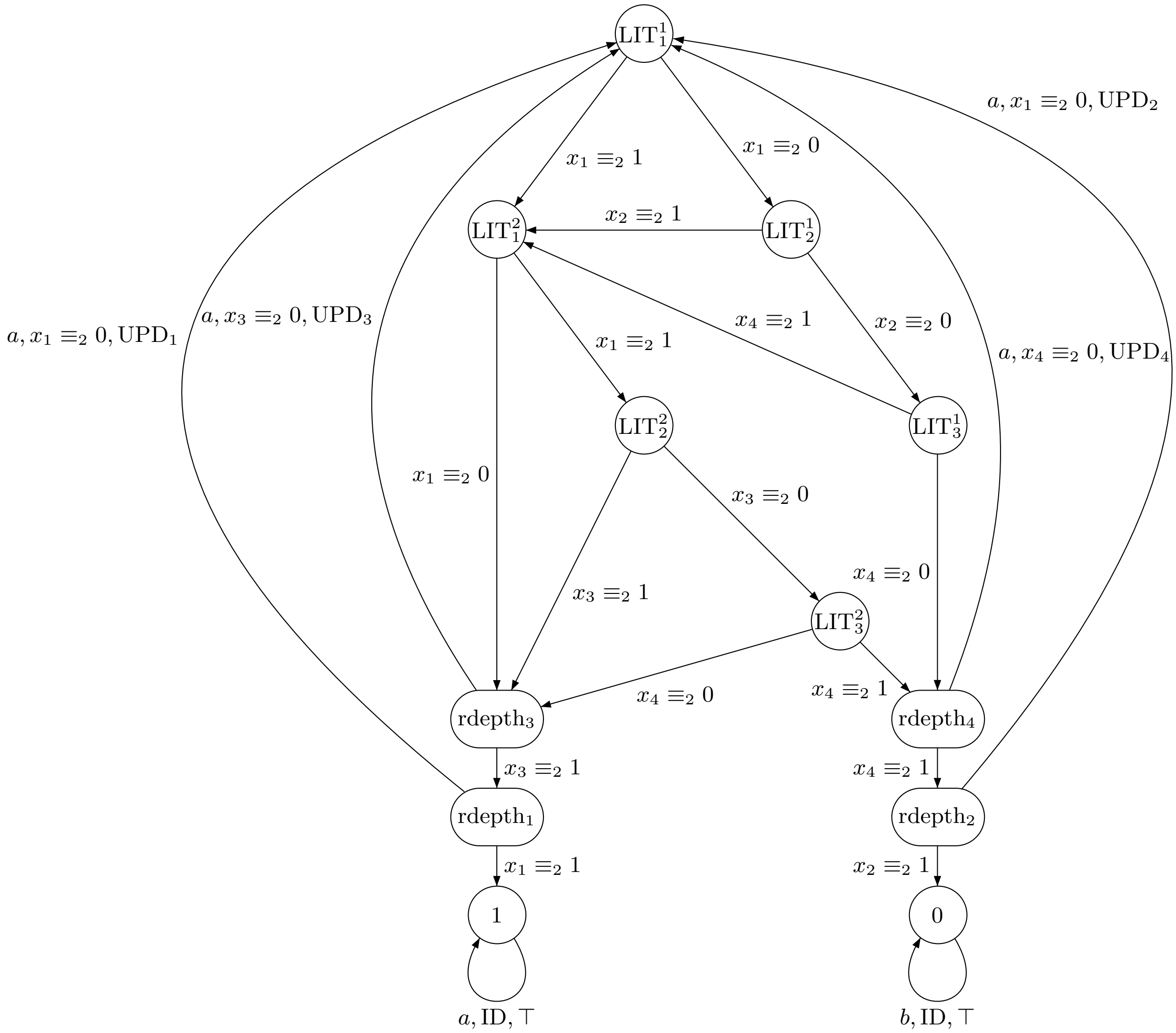


Fig. 6. ESSA associated with $\forall x_1 \exists x_2 \forall x_3 \exists x_4 (x_1 \vee x_2 \vee x_4) \wedge (\neg x_1 \vee x_3 \vee \neg x_4)$

input: an ESSA $\mathcal{A}$, $a \in \Sigma$ and $n, m \in \mathbb{N}$ (with a binary representation);
output: 1 if the nth occurrence of a in $w_{\mathcal{A}}$ is in position less than m. 0 otherwise.

The proof of Lemma 11 entails that this problem is also PSPACE-hard when n, m are encoded with a binary representation. Indeed, ϕ is not QBF satisfiable iff the first occurrence of b in $w_{\mathcal{A}}$ is in position less than $(2^{8\times|\phi|} \times 4 \times |\phi|^2) + 1$. The PSPACE upper bound of the occurrence problem is also easy to establish.

7 Concluding remarks

We have introduced a first-order theory of periodicity constraints IPC^{++} whose satisfiability is PSPACE-complete and a version of LTL with past whose atomic formulae are constraints from IPC^{++} (with comparison of variables at different states). PLTL$^{\text{mod}}$ is a very concise logical formalism to deal with periodicity constraints. We have shown that PLTL$^{\text{mod}}$ model-checking and satisfiability are PSPACE-complete and that PLTL$^{\exists}$(IPC^{+}), the fragment PLTL(IPC^{+}) of PLTL$^{\text{mod}}$ extended with the quantifier $\exists$ has problems in EXPSPACE. As an application, we have also proved that the equivalence problem for ESSA introduced in [LM01, Sect. 5] is PSPACE-complete, even if restricted to two variables.

In Table 1, we recall our main results about LTL and PLTL over periodicity constraints and we indicate open problems related to them. At the intersection of a line labeled by a constraint language L and a column labeled by the logical languages $\mathcal{L}/\mathcal{L}'$, we provide the decidability/complexity status of the satisfiability problem for the logics $\mathcal{L}$(L) and $\mathcal{L}'$(L), respectively. If there is no difference between LTL and PLTL, we provide a unique status. The question mark '?'

	LTL/PLTL	LTL/PLTL + $\downarrow$	LTL/PLTL + $\exists$
$\{x < y, x = y\}$	PSPACE-complete [DD02]	?/undecidable	undecidable
$\{x - y \geq c,\ c \geq 0\}$	undecidable	undecidable	undecidable
$\{x < y, x = y, x < c, x = c\}$	in EXPSPACE [DD03]	?/undecidable	undecidable
IPC + $\{x < y, x = y\}$	?	?/undecidable	undecidable
IPC^{+}	PSPACE-complete Theorem 5	in EXPSPACE Lemma 9	in EXPSPACE Lemma 9
IPC^{++}	PSPACE-complete Theorem 5	?	?

Table 1. Summary and open problems

in the table refers to the decidability status. All undecidability results are consequences of the fact that LTL over the constraint language allowing atomic constraint of the form $x = y$ and $x = y + 1$ is undecidable by simulation of two-counter machines[1].

Two families of open problems can be distinguished from Table 1:

1. How to deal with the constraint language IPC + $\{x < y, x = y\}$? The difficulty with the decidability status of PLTL(IPC + $\{x < y, x = y\}$) is that LTL($\{x < y, x = y\}$) already characterizes non ω-regular sequences of constraints, see details in [DD03].
2. How to deal with $\downarrow$ in the presence of atomic constraints of the form $x = y$? For instance, the decidability status of LTL($\{x = y\}$) + $\downarrow$ even restricted to formulae with a *unique local* variable is open. The formula $\Box \downarrow_{x'=x} \mathsf{X}\Box(x \neq x')$ belongs to this language.

References

[Aba89] M. Abadi. The power of temporal proofs. *Theoretical Computer Science*, 65:35–83, 1989.

[AD94] R. Alur and D. Dill. A theory of timed automata. *Theoretical Computer Science*, 126:183–235, 1994.

[AH93] R. Alur and Th. Henzinger. Real-time logics: complexity and expressiveness. *Information and Computation*, 104(1):35–77, 1993.

[AH94] R. Alur and Th. Henzinger. A really temporal logic. *Journal of the Association for Computing Machinery*, 41(1):181–204, 1994.

[BB03] C. Bartzis and T. Bultan. Efficient symbolic representation for arithmetic constraints in verification. *International Journal of Foundations of Computer Science*, 14(4):605–624, 2003.

[BBFS96] E. Bertino, C. Bettini, E. Ferrari, and P. Samarati. Supporting periodic authorizations and temporal reasoning in database access control. In *22nd VLDB, Bombay, India*, pages 472–483, 1996.

[1] For instance, the undecidability of PLTL + $\downarrow(\{x = y, x < y\})$ can be shown by adapting the proof of [DD03, Theorem 10.1] with the domain $\langle \mathbb{Z}, =, succ\rangle$. We modify the formula φ_M (used in the proof of [DD03, Theorem 10.1]) encoding the behaviour of the machine M. First, we introduce a new variable z and the formula $\Box(z < \mathsf{X}z)$ is a new conjunct of φ_M. In that way, $\sigma, 0 \models \Box(z < \mathsf{X}z)$ iff $\sigma(0)(z), \sigma(1)(z), \ldots$ is isomorphic to $\mathbb{N}$. Second, all the variables x occurring in φ_M are forced to take their values in $\{\sigma(i)(z) : i \in \mathbb{N}\}$, which can be done with the formula $\Box(\downarrow_{x'=x} \mathsf{F}^{-1}(z = x') \vee \mathsf{F}(z = x'))$. The conjunction of all these formulae is added to φ_M. Third, we replace every occurrence of $succ(\mathsf{X}^{n_1}x_1, \mathsf{X}^{n_2}x_2)$ in φ_M stating that the value of x_2 at the n_2th next state is 1 plus the value of x_1 at the n_1th next state, by

$$\downarrow_{x'_1=\mathsf{X}^{n_1}x_1} \downarrow_{x'_2=\mathsf{X}^{n_2}x_2} x'_1 < x'_2 \wedge \mathsf{Everywhere}((x'_1 < z) \Rightarrow ((x'_2 = z) \vee (x'_2 < z))),$$

with $\mathsf{Everywhere}(\psi) = \mathsf{G}^{-1}(\psi) \vee \Box(\psi)$. The quantification over $\mathbb{Z}$ that allows to express the successor function by means of $<$ is encoded by a quantification over the states, which is fine by the properties satisfied by the interpretation of z. The modified version of φ_M (with new conjuncts) encodes the behaviour of M, whence the undecidability of PLTL + $\downarrow(\{x = y, x < y\})$. The decidability status of LTL + $\downarrow(\{x = y, x < y\})$ (no past-time operators) is open.

[BBFS98] E. Bertino, C. Bettini, E. Ferrari, and P. Samarati. An access control model supporting periodicity constraints and temporal reasoning. *ACM Transactions on Databases Systems*, 23(3):231–285, 1998.

[BC02] Ph. Balbiani and J.F. Condotta. Computational complexity of propositional linear temporal logics based on qualitative spatial or temporal reasoning. In A. Armando, editor, *Frontiers of Combining Systems (FroCoS'02)*, volume 2309 of *Lecture Notes in Artificial Intelligence*, pages 162–173. Springer, Berlin, 2002.

[BDFP00] P. Bouyer, C. Dufourd, E. Fleury, and A. Petit. Are timed automata updatable? In *CAV'00*, volume 1855 of *Lecture Notes in Computer Science*, pages 464–479. Springer, Berlin, 2000.

[BEH95] A. Bouajjani, R. Echahed, and P. Habermehl. On the verification problem of nonregular properties for nonregular processes. In *LICS'95*, pages 123–133, 1995.

[Ber80] L. Berman. The complexity of logical theories. *Theoretical Computer Science*, 11:71–78, 1980.

[BH99] A. Bouajjani and P. Habermehl. Symbolic reachability analysis of FIFO-channel systems with nonregular sets of configurations. *Theoretical Computer Science*, 221(1–2):211–250, 1999.

[Bla00] P. Blackburn. Representation, reasoning, and relational structures: a hybrid logic manifesto. *Logic Journal of the IGPL*, 8(3):339–365, 2000.

[CC00] H. Comon and V. Cortier. Flatness is not a weakness. In *14 Int. Workshop Computer Science Logic*, volume 1862 of *Lecture Notes in Computer Science*, pages 262–276. Springer-Verlag, 2000.

[Čer94] K. Čerāns. Deciding properties of integral relational automata. In *ICALP-21*, volume 820 of *Lecture Notes in Computer Science*, pages 35–46. Springer, Berlin, 1994.

[CFP02] C. Combi, M. Franceschet, and A. Peron. A logical approach to represent and reason about calendars. In *Int. Symposium on Temporal Representation and Reasoning*, pages 134–140. IEEE Computer Society Press, 2002.

[CFP04] C. Combi, M. Franceschet, and A. Peron. Representing and reasoning about temporal granularities. *Journal of Logic and Computation*, 14(1):51–77, 2004.

[CJ98] H. Comon and Y. Jurski. Multiple counters automata, safety analysis and Presburger arithmetic. In A. Hu and M. Vardi, editors, *Proc. Computer Aided Verification, Vancouver*, volume 1427 of *Lecture Notes in Computer Science*, pages 268–279. Springer, Berlin, 1998.

[CM77] A. Chandra and P. Merlin. Optimal implementation of conjunctive queries in relational databases. In *9th ACM Symposium on Theory of Computing*, pages 77–90, 1977.

[CMP99] S. Cerrito, M. Cialdea Mayer, and S. Praud. First-order linear temporal logic over finite time structures. In H. Ganzinger, D. McAllester, and A. Voronkov, editors, *6th Int. Conference on Logic Programming and Automated Reasoning, Tbilisi, Republic of Georgia (LPAR'99)*, volume 1705 of *Lecture Notes in Computer Science*, pages 62–76. Springer, 1999.

[DD02] S. Demri and D. D'Souza. An automata-theoretic approach to constraint LTL. In M. Agrawal and A. Seth, editors, *FST&TCS'02, Kanpur*, volume 2556 of *Lecture Notes in Computer Science*, pages 121–132. Springer, Berlin, 2002.

[DD03] S. Demri and D. D'Souza. An automata-theoretic approach to constraint LTL. Technical Report LSV-03-11, LSV, August 2003. 40 pages. Extended version of [DD02] (submitted).

[Dem04] S. Demri. LTL over integer periodicity constraints (extended abstract). In *FOSSACS'04*, Lecture Notes in Computer Science. Springer, Berlin, 2004. to appear.

[DF99] R. Downey and M. Fellows. *Parameterized Complexity*. Monographs in Computer Science. Springer, 1999.

[DFS99] R. Downey, M. Fellows, and U. Stege. Computational tractability: The view from Mars. *Bulletin of EATCS*, 69:73–97, 1999.

[DPK03] Z. Dang, P. San Pietro, and R. Kemmerer. Presburger liveness verification of discrete timed automata. *Theoretical Computer Science*, 299:413–438, 2003.

[FL02] A. Finkel and J. Leroux. How to compose Presburger accelerations: Applications to broadcast protocols. In M. Agrawal and A. Seth, editors, *FST&TCS'02, Kanpur*, volume 2256 of *Lecture Notes in Computer Science*, pages 145–156. Springer, Berlin, 2002.

[FR74] M. Fischer and M. Rabin. Super-exponential complexity of Presburger arithmetic. In R. Karp, editor, *Complexity of Computation*, volume 7 of *SIAM-AMS proceedings*, pages 27–42. American Mathematical Society, 1974.

[FR75] J. Ferrante and Ch. Rackoff. A decision procedure for the first order theory of real addition with order. *SIAM Journal of Computing*, 4(1):69–76, 1975.

[FS00] A. Finkel and G. Sutre. Decidability of reachability problems for classes of two counters automata. In *17th Ann. Symp. Theoretical Aspects of Computer Science*, volume 2256 of *Lecture Notes in Computer Science*, pages 346–357. Springer-Verlag, 2000.

[GHR94] D. Gabbay, I. Hodkinson, and M. Reynolds. *Temporal Logic - Mathematical Foundations and Computational Aspects, Volume 1*. Oxford University Press, 1994.

[GK03] P. Gastin and D. Kuske. Satisfiability and model checking for MSO-definable temporal logics are in PSPACE. In *CONCUR'03*, volume 2761 of *Lecture Notes in Computer Science*, pages 222–236. Springer, 2003.

[GKK+03] D. Gabelaia, R. Kontchakov, A. Kurucz, F. Wolter, and M. Zakharyaschev. On the computational complexity of spatio-temporal logics. In *FLAIRS'03, St Augustine, Florida*, pages 460–464, 2003.

[GKWZ03] D. Gabbay, A. Kurucz, F. Wolter, and M. Zakharyaschev. *Many-dimensional modal logics: theory and practice*. Cambridge University Press, 2003.

[Gor96] V. Goranko. Hierarchies of modal and temporal logics with references pointers. *Journal of Logic, Language and Information*, 5:1–24, 1996.

[GS66] S. Ginsburg and E. Spanier. Semigroups, Presburger formulas and languages. *Pacific Journal of Mathematics*, 16(2):285–296, 1966.

[HMR03] Th. Henzinger, R. Majumdar, and J.F. Raskin. A classification of symbolic transitions systems. *ACM Transactions on Computational Logic*, 2003. to appear.

[HU79] J. Hopcroft and J. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 1979.

[JKMS02] P. Jančar, A. Kučera, F. Möller, and Z. Sawa. Equivalence-checking with one-counter automata: a generic method for proving lower bounds. In *FOS-*

SACS'02, volume 2256 of *Lecture Notes in Computer Science*, pages 172–186. Springer, Berlin, 2002.

[KK67] G. Kreisel and J.L. Krivine. *Elements of Mathematical Logic*. Studies in Logic and the Foundations of Mathematics. North-Holland Publishing Company, 1967.

[Kou94] M. Koubarakis. Complexity results for first-order theories of temporal constraints. In J. Doyle, E. Sandewall, and P. Torasso, editors, *4th International Conference on Principles of Knowledge Representation and Reasoning (KR'94)*, pages 379–390, 1994.

[Krö90] F. Kröger. On the interpretability of arithmetic in temporal logic. *Theoretical Computer Science*, 73:47–61, 1990.

[LM01] U. Dal Lago and A. Montanari. Calendars, time granularities, and automata. In *Int. Symposium on Spatial and Temporal Databases*, volume 2121 of *Lecture Notes in Computer Science*, pages 279–298. Springer, Berlin, 2001.

[LMP03] U. Dal Lago, A. Montanari, and G. Puppis. Towards compact and tractable automaton-based representations of time granularities. In *ICTCS 2003*, volume 2841 of *Lecture Notes in Computer Science*, pages 72–85. Springer, Berlin, 2003.

[LMS02] F. Laroussinie, N. Markey, and Ph. Schnoebelen. Temporal logic with forgettable past. In *LICS'02*, pages 383–392. IEEE Computer Society, 2002.

[Lut01] C. Lutz. Interval-based temporal reasoning with general TBoxes. In *17th International Joint Conference on Artificial Intelligence (IJCAI'01)*, pages 89–94. Morgan-Kaufmann, 2001.

[MOS04] M. Müller-Olm and H. Seidl. Program analysis through linear algebra. In *ACM Symposium on Principles of Programming Languages (POPL'04)*, 2004. to appear.

[NS92] M. Niezette and J. Stevenne. An efficient symbolic representation of periodic time. In *Proc. of the International Conference on Information and Knowledge Management, Baltimore, Maryland*, volume 752 of *Lecture Notes in Computer Science*, pages 161–168. Springer, 1992.

[Ohl94] H.J. Ohlbach. Calendar logic. In *[GHR94]*, chapter 19. 1994.

[Pre29] M. Presburger. Über die Vollständigkeit eines gewissen Systems der Arithmetik ganzer Zahlen, in welchem die Addition als einzige Operation hervortritt. *Comptes Rendus du premier congrès de mathématiciens des Pays Slaves, Warszawa*, pages 92–101, 1929.

[Rev02] P. Revesz. *Introduction to Constraint Databases.* Springer, New York, 2002.

[SC85] A. Sistla and E. Clarke. The complexity of propositional linear temporal logic. *Journal of the Association for Computing Machinery*, 32(3):733–749, 1985.

[Tar51] A. Tarski. *A Decision Method for Elementary Algebra and Geometry*. University of California Press, 1951.

[TC98] D. Toman and J. Chomicki. DATALOG with integer periodicity constraints. *Journal of Logic Programming*, 35(3):263–290, 1998.

[Tra63] B. Trahtenbrot. Impossibility of an algorithm for the decision problem in finite classes. *American Mathematical Society - Translation Series*, 23(2):1–5, 1963.

[vEB97] P. van Emde Boas. The convenience of tilings. In A. Sorbi, editor, *Complexity, Logic, and recursion Theory*, volume 187 of *Lecture Notes in Pure and Applied Logic*, pages 331–363. Marcel Dekker, Inc., 1997.

[VW94] M. Vardi and P. Wolper. Reasoning about infinite computations. *Information and Computation*, 115:1–37, 1994.

[WB00] P. Wolper and B. Boigelot. On the construction of automata from linear arithmetic constraints. In *6th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, volume 1785 of *Lecture Notes in Computer Science*, pages 1–19. Springer, Berlin, 2000.

[Wij00] J. Wijsen. A string based-model for infinite granularities. In *AAAI Workshop on Spatial and Temporal Granularity*, pages 9–16. AAAI Press, 2000.

[Wol01] P. Wolper. Representing periodic temporal information with automata. In *8th International Symposium on Temporal Representation and Reasoning (TIME'01)*, page 179. IEEE Computer Society, 2001.