Contents

1 Introduction 2

2 Preliminaries 5
   2.1 Language of constraints 5
   2.2 Syntax of CLTLB 6
   2.3 Semantics 6
   2.4 CLTL with automata 7
      2.4.1 Completion property 10

3 Satisfiability of CLTLB(D) without automata 12
   3.1 Bounded Satisfiability Problem 12
   3.2 Avoiding explicit symbolic valuations 13
   3.3 An encoding for BSP without automata 14
   3.4 Correctness of the BSP encoding 17

4 Bounded Satisfiability of CLTLB(IPC*) 23
   4.1 Simplifying the condition of existence of arithmetical models 33

5 Complexity and Completeness 36

6 Applications of k-bounded satisfiability 39

7 Related works 40

8 Conclusions and further developments 43
Constraint LTL Satisfiability Checking without Automata*

Marcello M. Bersani\textsuperscript{1}, Achille Frigeri\textsuperscript{1}, Angelo Morzenti\textsuperscript{1}, Matteo Pradella\textsuperscript{1}, Matteo Rossi\textsuperscript{1}, and Pierluigi San Pietro\textsuperscript{1}

\textsuperscript{1}Politecnico di Milano - DEIB

May 1, 2014

Abstract

This paper introduces a novel technique to decide the satisfiability of formulæ written in the language of Linear Temporal Logic with both future and past operators and atomic formulæ belonging to constraint system $\mathcal{D}$ (CLTL$\mathcal{D}$ for short). The technique is based on the concept of bounded satisfiability, and hinges on an encoding of CLTL$\mathcal{D}$ formulæ into QF-EUD, the theory of quantifier-free equality and uninterpreted functions combined with $\mathcal{D}$. Similarly to standard LTL, where bounded model-checking and SAT-solvers can be used as an alternative to automata-theoretic approaches to model-checking, our approach allows users to solve the satisfiability problem for CLTL$\mathcal{D}$ formulæ through SMT-solving techniques, rather than by checking the emptiness of the language of a suitable automaton. The technique is effective, and it has been implemented in our Zot formal verification tool.

1 Introduction

Finite-state system verification has attained great successes, both using automata-based and logic-based techniques. Examples of the former are the so-called explicit-state model checkers \cite{holzmann1997} and symbolic model checkers \cite{clarke1996}. However, some of the best results in practice have been obtained by logic-based techniques, such as Bounded Model Checking (BMC) \cite{biere1999}. In BMC, a finite-state machine $A$ (typically, a version of Büchi automata) and a desired property $P$ expressed in Propositional Linear Temporal Logic (PLTL) are translated into a Boolean formula $\phi$ to be fed to a SAT solver. The translation is made finite by bounding the number of time instants. However, infinite behaviors, which are crucial in proving, e.g., liveness properties, are also considered by using the well-known property that a

\*This research was partially supported by Programme IDEAS-ERC and Project 227977-SMScom.
Büchi automaton accepts an infinite behavior if, and only if, it accepts an infinite periodic behavior. Hence, chosen a bound $k > 0$, a Boolean formula $\phi_k$ is built, such that $\phi_k$ is satisfiable if and only if there exists an infinite periodic behavior of the form $\alpha\beta^\omega$, with $|\alpha\beta| \leq k$, that is compatible with system $A$ while violating property $P$. This procedure allows counterexample detection, but it is not complete, since the violations of property $P$ requiring “longer” behaviors, i.e., of the form $\alpha\beta^\omega$ with $|\alpha\beta| > k$, are not detected. However, in many practical cases it is possible to find bounds large enough for representing counterexamples, but small enough so that the SAT solver can actually find them in a reasonable time.

Clearly, the BMC procedure can be used to check satisfiability of a PLTL formula, without considering a finite state system $A$. This has practical applications, since a PLTL formula can represent both the system and the property to be checked (see, e.g., Pradella et al. [2013], where the translation into Boolean formulae is made more specific for dealing with satisfiability checking and metric temporal operators). We call this case Bounded Satisfiability Checking (BSC), which consists in solving a so-called Bounded Satisfiability Problem: Given a PLTL formula $P$, and chosen a bound $k > 0$, define a Boolean formula $\phi_k$ such that $\phi_k$ is satisfiable if, and only if, there exists an infinite periodic behavior of the form $\alpha\beta^\omega$, with $|\alpha\beta| \leq k$, that satisfies $P$.

More recently, great attention has been given to the automated verification of infinite-state systems. In particular, many extensions of temporal logic and automata have been proposed, typically by adding integer variables and arithmetic constraints. For instance, PLTL has been extended to allow formulae with various kinds of arithmetic constraints Comon and Cortier (2000); Demandri and D’Souza (2002). This has led to the study of CLTL($D$), a general framework extending the future-only fragment of PLTL by allowing arithmetic constraints belonging to a generic constraint system $D$. The resulting logics are expressive and well-suited to define infinite-state systems and their properties, but, even for the bounded case, their satisfiability is typically undecidable Demri and Gascon (2006), since they can simulate general two-counter machines when $D$ is powerful enough (e.g., Difference Logic).

However, there are some decidability results, which allow in principle for some kind of automatic verification. Most notably, satisfiability of CLTL($D$) is decidable (in PSPACE) when $D$ is the class of Integer Periodic Constraints (IPC*) Demri and Gascon (2007), or when it is the structure $(D, <, =)$ with $D \in \{\mathbb{N}, \mathbb{Z}, \mathbb{Q}, \mathbb{R}\}$ Demri and D’Souza (2007). In these cases, decidability is shown by using an automata-based approach similar to the standard case for LTL, by reducing satisfiability checking to the verification of the emptiness of Büchi automata. Given a CLTL($D$) formula $\phi$, with $D$ as in the above cases, it is possible to define an automaton $A_\phi$ such that $\phi$ is satisfiable if, and only if, the language recognized by $A_\phi$ is not empty.

These results, although of great theoretical interest, are of limited practical relevance for what concerns a possible implementation, since the involved constructions are very inefficient, as they rely on the complementation of Büchi automata.

In this paper, we extend the above results to a more general logic, called CLTLB($D$), which is an extension of PLTLB (PLTL with Both future and past operators) with arithmetic constraints in constraint system $D$, and define a procedure for satisfiability checking that does not rely on automata constructions.

The idea of the procedure is to determine satisfiability by checking a finite number
of $k$-satisfiability problems. Informally, $k$-satisfiability amounts to looking for ultimately periodic symbolic models of the form $\alpha \beta \omega$, i.e., such that prefix $\alpha \beta$ of length $k$ admits a bounded arithmetic model (up to instant $k$). Although the $k$-bounded problem is defined with respect to a bounded arithmetical model, it provides a representation of infinite symbolic models by means of ultimately periodic words. When CLTLB($D$) has the property that its ultimately periodic symbolic models, of the form $\alpha \beta \omega$, always admit an arithmetic model, then the $k$-satisfiability problem can be reduced to satisfiability of QF-EU$_D$ (the theory of quantifier-free equality and uninterpreted functions combined with $D$). In this case, $k$-satisfiability is equivalent to satisfiability over infinite models.

There are important examples of constraint systems $D$, such as for example IPC$^*$, in which determining the existence of arithmetical models is achieved by complementing a Büchi automaton $A_C$. In this paper we define a novel condition, tailored to ultimately periodic models of the form $\alpha \beta \omega$, which is proved to be equivalent to the one captured by automaton $A_C$. Thanks to this condition, checking for the existence of arithmetical models can be done in a bounded way, without resorting to the construction (and the complementation) of Büchi automata. This is the key result that makes our decision procedure applicable in practice.

Symmetrically to standard LTL, where bounded model-checking and SAT-solvers can be used as an alternative to automata-theoretic approaches to model-checking, reducing satisfiability to $k$-satisfiability allows us to determine the satisfiability of CLTLB($D$) formulae through Satisfiability Modulo Theories (SMT) solvers, instead of checking the emptiness of a Büchi automaton. Moreover, when the length of all prefixes $\alpha \beta$ to be tested is bounded by some $K \in \mathbb{N}$, then the number of bounded problems to be solved is finite. Therefore, we also prove that $k$-satisfiability is complete with respect to the satisfiability problem, i.e., by checking at most $K$ bounded problems the satisfiability of CLTLB($D$) formulae can always be determined.

To the best of our knowledge, our results provide the first effective implementation of a procedure for solving the CLTLB($D$) satisfiability problem: we show that the encoding into QF-EU$_D$ is linear in the size of the formula to be checked and quadratic in the length $k$. The procedure is implemented in the Zot toolkit\footnote{http://zot.googlecode.com} which relies on standard SMT-solvers, such as Z3 [Microsoft Research (2009)].

The paper is organized as follows. Section 2 describes CLTL($D$) and CLTLB($D$), and their main known decidability results and techniques. Section 3 defines the $k$-satisfiability problem, introduces the bounded encoding of CLTLB($D$) formulae, and shows its correctness. Section 4 introduces a novel, bounded condition for checking the satisfiability of CLTLB($D$) formulae when $D$ is IPC$^*$, and discusses some cases under which the encoding can be simplified. Section 5 studies the complexity of the defined encoding and proves that, provided that $D$ satisfies suitable conditions, there exists a completeness threshold. Section 6 illustrates an application of the CLTLB logic and the Zot toolkit to specify and verify a system behavior. Section 7 describes relevant related works. Finally, Section 8 concludes the paper highlighting some possible applications of the implemented decision procedure for CLTLB($D$).
2 Preliminaries

This section presents an extension to Kamp’s (1968) PLTLB, by allowing formulae over a constraint system. As suggested in Comon and Cortier (2000), and unlike the approach of Demri (2004), the propositional variables of this logic are Boolean terms or atomic arithmetic constraints.

2.1 Language of constraints

Let \( V \) be a finite set of variables; a constraint system is a pair \( D = (D, R) \) where \( D \) is a specific domain of interpretation for variables and constants and \( R \) is a family of relations on \( D \). An atomic \( D \)-constraint is a term of the form \( R(x_1, \ldots, x_n) \), where \( R \) is an \( n \)-ary relation of \( R \) on domain \( D \) and \( x_1, \ldots, x_n \) are variables. A \( D \)-valuation is a mapping \( v : V \to D \), i.e., an assignment of a value in \( D \) to each variable. A constraint is satisfied by a \( D \)-valuation \( v \), written \( v \models D R(x_1, \ldots, x_n) \), if \( (v(x_1), \ldots, v(x_n)) \in R \).

In Section 4 we consider \( D \) to be Integer Periodic Constraints (IPC\(^*\)) or its fragments (e.g., \((\mathbb{Z}, <, =)\) or \((\mathbb{N}, <, =)\)) and \((D, <, =)\) when \(< \) is a dense order without endpoints, e.g., \( D \in \{\mathbb{R}, \mathbb{Q}\} \). The language IPC\(^*\) is defined by the following grammar, where \( \xi \) is the axiom:

\[
\xi := \theta \mid x < y \mid \xi \land \xi \mid \neg \xi
\]

\[
\theta := x \equiv_c d \mid x \equiv_c y + d \mid x = y \mid x < d \mid x = d \mid \theta \land \theta \mid \neg \theta
\]

where \( x, y \in V, c \in \mathbb{N}^+ \) and \( d \in \mathbb{Z} \). The first definition of IPC\(^*\) can be found in Demri and Gascon (2005); it is different from ours since it allows existentially quantified formulae (i.e., \( \theta := \exists x \theta \)) to be part of the language. However, since IPC\(^*\) is a fragment of Presburger arithmetic, it has the same expressivity as the above quantifier-free version (but with an exponential blow-up to remove quantifiers).

Given a valuation \( v \), the satisfaction relation \( \models D \) is defined:

- \( v \models D x \sim y \) iff \( v(x) \sim v(y) \);
- \( v \models D x \sim d \) iff \( v(x) \sim d \);
- \( v \models D x \equiv_c d \) iff \( v(x) - d = kc \) for some \( k \in \mathbb{Z} \);
- \( v \models D x \equiv_c y + d \) iff \( v(x) - v(y) - d = kc \) for some \( k \in \mathbb{Z} \);
- \( v \models D \xi_1 \land \xi_2 \) iff \( v \models D \xi_1 \) and \( v \models D \xi_2 \);
- \( v \models D \neg \xi \) iff \( v \not\models D \xi \);

where \( \sim \) is either \( = \) or \(< \). A constraint is satisfiable if there exists a valuation \( v \) such that \( v \models D \xi \). Given a set of IPC\(^*\) constraints \( C \), we write \( v \models D C \) when \( v \models D \xi \) for every \( \xi \in C \).
2.2 Syntax of CLTLB

CLTLB(\(D\)) is defined as an extension of PLTLB, where atomic formulae are relations from \(\mathcal{R}\) over arithmetic temporal terms defined in \(D\). The resulting logic is actually equivalent to the quantifier-free fragment of first-order LTL over signature \(\mathcal{R}\). Let \(x\) be a variable; arithmetic temporal terms (a.t.t.) are defined as:

\[
\alpha := c | x | X\alpha | Y\alpha.
\]

where \(c\) is a constant in \(D\) and \(x\) is a variable over \(D\). The syntax of (well formed) formulae of CLTLB(\(D\)) is recursively defined as follows:

\[
\phi := R(\alpha_1, \ldots, \alpha_n) | \phi \land \phi | \neg \phi | X\phi | Y\phi | \phi U \phi | \phi S \phi
\]

where \(\alpha_i\)'s are a.t.t.'s, \(R \in \mathcal{R}\); \(X, Y, U,\) and \(S\) are the usual “next”, “previous”, “until”, and “since” operators from LTL.

Note that \(X\) and \(X\) are two distinct operators; if \(\phi\) is a formula, \(X\phi\) has the standard PLTL meaning, while \(X\alpha\) denotes the value of a.t.t. \(\alpha\) in the next time instant. Thanks to the obvious property that \(XYx \equiv YXx \equiv x\), we will assume, with no loss of generality, that a.t.t.'s do not contain any nested alternated occurrences of the operators \(X\) and \(Y\). Each relation symbol is associated with a natural number denoting its arity. As we will see in Section 3.4, we can treat separately 0-ary relations, i.e., propositional letters, whose set is denoted by \(\mathcal{R}_0\). We also write CLTLB(\(D, \mathcal{R}_0\)) to denote the language CLTLB over the constraint system \(D\) whose 0-ary relations are exactly those in \(\mathcal{R}_0\).

CLTL(\(D\)) is the future-only fragment of CLTLB(\(D\)).

The depth \(|\alpha|\) of an a.t.t. is the total amount of temporal shift needed in evaluating \(\alpha\):

\[
|x| = 0, \quad |X\alpha| = |\alpha| + 1, \quad |Y\alpha| = |\alpha| - 1.
\]

Let \(\phi\) be a CLTLB(\(D, \mathcal{R}_0\)) formula, \(x\) a variable of \(V\) and \(\Gamma_x(\phi)\) the set of all a.t.t.'s occurring in \(\phi\) in which \(x\) appears. We define the “look-forwards” \([\phi]_x\) and “look-backwards” \([\phi]_x\) of \(\phi\) relatively to \(x\) as:

\[
[\phi]_x = \max_{\alpha_i \in \Gamma_x(\phi)} \{0, |\alpha_i|\}, \quad [\phi]_x = \min_{\alpha_i \in \Gamma_x(\phi)} \{0, |\alpha_i|\}.
\]

The definitions above naturally extend to \(V\) by letting \([\phi] = \max_{x \in V} \{[\phi]_x\}, \quad [\phi] = \min_{x \in V} \{[\phi]_x\}\). Hence, \([\phi] ([\phi])\) is the largest (smallest) depth of all the a.t.t.'s of \(\phi\), representing the length of the future (past) segment needed to evaluate \(\phi\) in the current instant.

2.3 Semantics

The semantics of CLTLB(\(D, \mathcal{R}_0\)) formulae is defined with respect to a strict linear order representing time (\(\mathbb{Z}, <\)). Truth values of propositions in \(\mathcal{R}_0\), and values of variables belonging to \(V\) are defined by a pair \((\pi, \sigma)\) where \(\sigma : \mathbb{Z} \times V \to D\) is a function which defines the value of variables at each position in \(\mathbb{Z}\) and \(\pi : \mathbb{Z} \to \varphi(\mathcal{R}_0)\).
is a function associating a subset of the set of propositions with each element of $\mathbb{Z}$. Function $\sigma$ is extended to terms as follows:

$$\sigma(i, \alpha) = \sigma(i + |\alpha|, x_\alpha)$$

where $x_\alpha$ is the variable in $V$ occurring in term $\alpha$, if any; otherwise $x_\alpha = \alpha$. The semantics of a CLTLB($D$, $\mathcal{R}_0$) formula $\phi$ at instant $i \geq 0$ over a linear structure $(\pi, \sigma)$ is recursively defined by means of a satisfaction relation $|$ as follows, for every formulae $\phi, \psi$ and for every a.t.t. $\alpha$:

$$(\pi, \sigma), i \models p \text{ iff } p \in \pi(i) \text{ for } p \in \mathcal{R}_0$$

$$(\pi, \sigma), i \models R(\alpha_1, \ldots, \alpha_n) \text{ iff } (\sigma(i + |\alpha_1|, x_{\alpha_1}), \ldots, \sigma(i + |\alpha_n|, x_{\alpha_n})) \in R \text{ for } R \in \mathcal{R} \setminus \mathcal{R}_0$$

$$(\pi, \sigma), i \models \neg \phi \text{ iff } (\pi, \sigma), i \not\models \phi$$

$$(\pi, \sigma), i \models \phi \land \psi \text{ iff } (\pi, \sigma), i \models \phi \text{ and } (\pi, \sigma), i \models \psi$$

$$(\pi, \sigma), i \models X\phi \text{ iff } (\pi, \sigma), i + 1 \models \phi$$

$$(\pi, \sigma), i \models Y\phi \text{ iff } (\pi, \sigma), i - 1 \models \phi \land i > 0$$

$$(\pi, \sigma), i \models \phi U\psi \text{ iff } \exists j \geq i : (\pi, \sigma), j \models \psi \land (\pi, \sigma), n \models \psi \forall n : i < n < j$$

$$(\pi, \sigma), i \models \phi S\psi \text{ iff } \exists 0 \leq j \leq i : (\pi, \sigma), j \models \psi \land (\pi, \sigma), n \models \psi \forall n : j < n \leq i.$$
**Definition 1.** A symbolic valuation \(sv\) for \(\phi\) is a maximally consistent set of \(D\)-constraints over \(\text{terms}(\phi)\) and \(\text{const}(\phi)\).

The original definition of symbolic valuation for IPC* constraint systems in [Demri and Gascon (2005)] is slightly different. Our definition does not consider explicitly relation \(x = d\) and periodic relation \(x \equiv_c d\), with \(c, d \in D\), because they are inherently represented in the \(k\)-bounded arithmetical models defined in Section 3.1. Equality between variables and constants do not require to be symbolically represented by a symbolic constraint of the form \(x = d\) as \(k\)-bounded arithmetical models associate each variable with an “explicit” value from \(D\). Moreover, given \(x\) a value from \(D\), relation \(x \equiv_c d\) is inherently defined.

The satisfiability of a symbolic valuation is defined as follows, by considering each a.t.t. as a new fresh variable.

**Definition 2.** The set of all symbolic valuations for \(\phi\) is denoted by \(SV(\phi)\). Let \(A\) be a set of variables and fresh : \(\text{terms}(\phi) \rightarrow A\) be an injective function mapping each a.t.t of \(\phi\) to a fresh variable in set \(A\). Function fresh is naturally extended to every symbolic valuation \(sv\) for \(\phi\), by replacing each a.t.t. \(\alpha \in \text{terms}(\phi)\) in \(sv\) with fresh(\(\alpha\)). Symbolic valuations for \(\phi\) are now defined over the set fresh(\(\text{terms}(\phi)\)). A symbolic valuation \(sv\) for \(\phi\) is satisfiable if there exists a \(D\)-valuation \(v' : A \rightarrow D\), such that \(v' \models_D\) fresh(\(sv\)), i.e., satisfiability of \(sv\) considers all a.t.t.’s as fresh variables.

Given a symbolic valuation \(sv\) and a \(D\)-constraint \(\xi\) over a.t.t.’s, we write \(sv \models \xi\) if for every \(D\)-valuation \(v'\) such that \(v' \models_D\) fresh(\(sv\)) then we have \(v' \models_D\) fresh(\(\xi\)). We assume that the problem of checking \(sv \models \xi\) is decidable. The satisfaction relation \(\models\) can also be extended to infinite sequences \(\rho : \mathbb{N} \rightarrow SV(\phi)\) (or, equivalently, \(\rho \in SV(\phi)^\omega\)) of symbolic valuations; it is the same as \(\models\) for all temporal operators except for atomic formulae:

\[
\rho, i \models \xi\text{ iff }\rho(\(i\)) \models \xi
\]

Then, given a CLTLB(\(D\)) formula \(\phi\), we say that a symbolic model \(\rho\) symbolically satisfies \(\phi\) (or \(\rho\) is a symbolic model for \(\phi\)) when \(\rho, 0 \models \phi\).

In the rest of this section we consider CLTLB(\(D\)) formulae that do not include arithmetic temporal operator \(Y\). This is without loss of generality, as Property 3 will show.

**Definition 3.** A pair of symbolic valuations \((sv_1, sv_2)\) for \(\phi\) is locally consistent if, for all \(R \in D\):

\[
R(\(X^{i_1}x_1, \ldots, X^{i_n}x_n\)) \in sv_1 \text{ implies } R(\(X^{i_1-1}x_1, \ldots, X^{i_n-1}x_n\)) \in sv_2
\]

with \(i_j \geq 1\) for all \(j \in [1, n]\). A sequence of symbolic valuations \(sv_0, sv_1, \ldots\) is locally consistent if all pairs \((sv_i, sv_{i+1})\), \(i \geq 0\), are locally consistent.

A locally consistent infinite sequence \(\rho\) of symbolic valuations admits an arithmetic model, if there exists a \(D\)-valuation sequence \(\sigma\) such that \(\sigma, i \models \rho(\(i\))\), for all \(i \geq 0\). In this case, we write \(\sigma, 0 \models \rho\).

The following fundamental proposition draws a link between the satisfiability by sequences of symbolic valuations and by sequences of \(D\)-valuations.
Proposition 1 (Demri and D’Souza (2007)). A CLTL(D) formula \(\phi\) is satisfiable if, and only if, there exists a symbolic model for \(\phi\) which admits an arithmetical model, i.e., there exist \(\rho\) and \(\sigma\) such that \(\rho, 0 \models_{\text{sym}} \phi\) and \(\sigma, 0 \models \rho\).

Following Demri and D’Souza (2007), for constraint systems of the form \((D, <, \sim)\), where \(<\) is a strict total ordering on \(D\), it is possible to represent a symbolic valuation \(sv\) by its labeled directed graph \(G_{sv} = (V, \tau)\), \(\tau \subseteq V \times \{<, \sim\} \times V\), such that \((x, \sim, y) \in \tau\) if, and only if, \(x \sim y \in sv\). This construction extends also to sequences: given a sequence \(\rho\) of symbolic valuations, it is possible to represent \(\rho\) via the graph \(G_{\rho}\) obtained by superimposition of the graphs corresponding to the symbolic evaluations \(\rho(i)\). More formally \(G_{\rho} = (V \times \mathbb{N}, \tau_{\rho})\), where \(((x, i), \sim, (y, j)) \in \tau_{\rho}\) if, and only if, either \(i \leq j\) and \((x \sim y) \in \rho(i)\), or \(i > j\) and \((X^{i-j} x \sim y) \in \rho(j)\).

An infinite path \(d : \mathbb{N} \to V \times \mathbb{N}\) in \(G_{\rho}\), is called a forward (resp. backward) path if:

1. for all \(i \in \mathbb{N}\), there is an edge from \(d(i)\) to \(d(i + 1)\) (resp., an edge from \(d(i + 1)\) to \(d(i)\));
2. for all \(i \in \mathbb{N}\), if \(d(i) = (x, j)\) and \(d(i + 1) = (x', j')\), then \(j \leq j'\).

A forward (resp. backward) path is strict if there exist infinitely many \(i\) for which there is a <-labeled edge from \(d(i)\) to \(d(i + 1)\) (resp., from \(d(i + 1)\) to \(d(i)\)). Intuitively, a (strict) forward path represents a sequence of (strict) monotonic increasing values whereas a (strict) backward path represents a sequence of (strict) monotonic decreasing values.

Given a CLTL(D) formula \(\phi\), it is possible Demri and D’Souza (2007) to define a Büchi automaton \(A_{\phi}\) recognizing symbolic models of \(\phi\), and then reducing the satisfiability of \(\phi\) to the emptiness of \(A_{\phi}\). The idea is that automaton \(A_{\phi}\) should accept the intersection of the following languages, which defines exactly the language of symbolic models of \(\phi\):

1. the language of LTL models \(\rho\);
2. the language of sequences of locally consistent symbolic valuations;
3. the language of sequences of symbolic valuations which admit an arithmetical model.

Language (1) is accepted by the Vardi-Wolper automaton \(A_{\phi}\) of \(\phi\) Vardi and Wolper (1986), while language (2) is recognized by the automaton \(A_{\ell} = (SV(\phi), sv_0, \rightarrow, SV(\phi))\), where the states are \(SV(\phi)\), all accepting; \(sv_0\) is the initial state; and the transition relation is such that \(sv_i \rightarrow sv_{i+1}\) if, and only if, all pairs \((sv_i, sv_{i+1})\) are locally consistent Demri and D’Souza (2007).

If the constraint system we are considering has the completion property (defined next), then all sequences of locally consistent symbolic valuations admit an arithmetic model, and condition (3) reduces to (2).
2.4.1 Completion property

Each automaton involved in the definition of $A_\phi$ has the function of “filtering” sequences of symbolic valuations so that 1) they are locally consistent, 2) they satisfy an LTL property and 3) they admit a (arithmetic) model. For some constraint systems, admitting a model is a consequence of local consistency. A set of relations over $D$ has the completion property if, given:

(i) a symbolic valuation $sv$ over a finite set of variables $H \subseteq V,$

(ii) a subset $H' \subseteq H,$

(iii) a valuation $v'$ over $H'$ such that $v' \models sv'$, where $sv'$ is the subset of atomic formulae in $sv$ which uses only variables in $H'$

then there exists a valuation $v$ over $V$ extending $v'$ such that $v \models sv$. An example of such a relational structure is $(\mathbb{R}, <, =)$. Let $(D, <, =)$ be a relational structure defining the language of atomic formulae. We say that $D$ is dense, with respect to the order $<$, if for each $d, d' \in D$ such that $d < d'$, there exists $d'' \in D$ such that $d < d'' < d'$, whereas $D$ is said to be open when for each $d \in D$, there exist two elements $d', d'' \in D$ such that $d' < d < d''$.


**Lemma 1** (Lemma 5.3, Demri and D’Souza (2007)). Let $(D, <, =)$ be a relational structure where $D$ is infinite and $<$ is a total order. Then, it satisfies the completion property if, and only if, domain $D$ is dense and open.

The following result relies on the fact that every locally consistent sequence of symbolic valuations with respect to the relational structure $D$ admits a model.

**Proposition 2.** Let $\mathcal{D}$ be a relational structure satisfying the completion property and $\phi$ be a CLTL($\mathcal{D}$) formula. Then, the language of sequences of symbolic valuations which admit a model is $\omega$-regular.

In this case the automaton $A_\phi$ that recognizes exactly all the sequences of symbolic valuations which are symbolic models of $\phi$ is defined by the intersection (à la Büchi) $A_\phi = A_s \cap A_\ell$.

In general, however, language (3) may not be $\omega$-regular. Nevertheless, if the constraint system is of the form $(D, <, =)$, it is possible to define an automaton $A_C$ that accepts a superset of language (3), but such that all its ultimately periodic words are sequences of symbolic valuations that admit an arithmetic model. Actually, $A_C$ recognizes a sequence $\rho$ of symbolic valuations that satisfies the following property:

**Property 1.** There do not exist vertices $u$ and $v$ in the same symbolic valuation in $G_\rho$ satisfying all the following conditions:

1. there is an infinite forward path $d$ from $u$;
2. there is an infinite backward path $e$ from $v$;
3. $d$ or $e$ are strict;
4. for each \(i, j \in \mathbb{N}\), whenever \(d(i)\) and \(e(j)\) belong to the same symbolic valuation, there exists an edge, labeled by \(<\), from \(d(i)\) to \(e(j)\).

Informally, Property 1 guarantees that in the model there does not exist an infinite forward path whose values are infinitely often less than values of an infinite backward path; in other words, an infinite strict/non-strict monotonic increasing sequence of values can not be infinitely often less than an infinite non-strict/strict monotonic decreasing sequence of values.

The proposed method is general and it can be used whenever it is possible to build an automaton \(A_C\) which defines a condition \(C\) guaranteeing the existence of a sequence \(\sigma\) such that \(\sigma, 0 \models \rho\). In particular, for constraint systems IPC\(^*\), \((\mathbb{N}, <, =)\), and \((\mathbb{Z}, <, =)\), \(A_C\) can be effectively built. Let \(A_\phi\) be defined as the (Büchi) product of \(A_C, A_s, A_\ell\); since emptiness of Büchi automata can be checked just on ultimately periodic words, the language of \(A_\phi\) is empty if, and only if, \(\phi\) does not have a symbolic model (which is equivalent to not having an arithmetical model).

When the condition \(C\) is sufficient and necessary for the existence of models \(\sigma\) such that \(\sigma, 0 \models \rho\), then automaton \(A_\phi\) represents all the sequences of symbolic valuations which admit a model \(\sigma\). A fundamental lemma, on which Proposition 3 below relies, draws a sufficient and necessary condition for the existence of models of sequences of symbolic valuations.

**Lemma 2 (Demri and D’Souza (2007)).** Let \(\rho\) be an ultimately periodic sequence of symbolic valuations of the form \(\rho = \alpha\beta^\omega \in SV(\phi)^\omega\) that is locally consistent. Then, \(\sigma, 0 \models \rho\) (i.e., \(\rho\) admits a model \(\sigma\)) if, and only if, \(\rho\) satisfies \(C\).

Therefore, the satisfiability problem can be solved by checking the emptiness of the language recognized by the automaton \(A_\phi\).

**Proposition 3 (Demri and D’Souza (2007)).** A CLTL\((D)\) formula \(\phi\) is satisfiable if, and only if, the language \(L(A_\phi)\) is not empty.

In the next section, we provide a way for checking the satisfiability of CLTLB\((D)\) formulae that does not require the construction of automata \(A_\ell, A_s\) and \(A_C\). Our approach takes advantage of the semantics of CLTLB\((D)\) for building models of formulae through a semi-symbolic construction. We use a reduction to a Satisfiability Modulo Theories (SMT) problem which extends the one proposed for Bounded Model Checking [Biere et al. (2003)]. In the automata-based construction the definition of automaton \(A_\phi\) may be prohibitive in practice and requires to devise alternative ways that avoid the exhaustive enumeration of all the states in \(A_\phi\). In fact, the size of \(A_s\) is exponential with respect to the size of the formula and condition \(C\), which needs to be checked when the constraint system does not have the completion property, as in the case of \((\mathbb{Z}, <, =)\), is defined by complementing, through Safra’s algorithm, automaton \(A_{\neg C}\) which recognizes symbolic sequences satisfying the negated condition \(C\) [Demri and D’Souza (2007)]. Since to show the satisfiability of a formula one can exhibit an ultimately periodic model whose length may be much smaller than the size of automaton \(A_\phi\), in many cases the whole construction of \(A_\phi\) is useless. However, proving unsatisfiability is comparable in complexity to defining the whole automaton \(A_\phi\) because it requires to verify that no ultimately periodic model \(\alpha\beta^\omega\) can be constructed for size...
$\alpha\beta$ equal to the size of automaton $A_\phi$. Motivated by the arguments above, we define the bounded satisfiability problem which consists in looking for ultimately periodic symbolic models $\alpha\beta\omega$ such that prefix $\alpha\beta$ is of fixed length (which is an input of the problem) and which admits a finite arithmetical model $\sigma_k$. Since symbolic valuations partition the space of variable valuations, an assignment of values to terms uniquely identifies a symbolic valuation (see next Lemma 3). For this reason, we do not need to precompute the set $SV(\phi)$ and instead we enforce the periodicity between a pair of sets of relations, those defining the first and last symbolic valuations in $\beta$. We show that, when a formula $\phi$ is boundedly satisfiable, then it is also satisfiable. We provide a (polynomial-space) reduction from the bounded satisfiability problem to the satisfiability of formulae in the quantifier-free theory of equality and uninterpreted functions QF-EUF combined with $D$.

3 Satisfiability of CLTLB($D$) without automata

In this section, we introduce our novel technique to solve the satisfiability problem of CLTLB($D$) formulae without resorting to an automata-theoretic construction.

First, we provide the definition of the $k$-satisfiability problem for CLTLB($D$) formulae in terms of the existence of a so-called $k$-bounded arithmetical model $\sigma_k$, which provides a finite representation of infinite symbolic models by means of ultimately periodic words. This allows us to prove that $k$-satisfiability is still representative of the satisfiability problem as defined in Section 2.3. In fact, for some constraint systems, a bounded solution can be used to build the infinite model $\sigma$ for the formula from the $k$-bounded one $\sigma_k$ and from its symbolic model. We show in Section 3.4 that a formula $\phi$ is satisfiable if, and only if, it is $k$-satisfiable and its bounded solution $\sigma_k$ can be used to derive its infinite model $\sigma$. In case of negative answer to a $k$-bounded instance, we can not immediately entail the unsatisfiability of the formula. However, we prove in Section 3.5 that for every formula $\phi$ there exists an upper bound $K$, which can effectively be determined, such that if $\phi$ is not $k$-satisfiable for all $k$ in $[1,K]$ then $\phi$ is unsatisfiable.

3.1 Bounded Satisfiability Problem

We first define the Bounded Satisfiability Problem (BSP), by considering bounded symbolic models of CLTLB($D$) formulae. For simplicity, we consider the set $R_0$ of propositional letters to be empty; this is without loss of generality, as Property 2 of Section 3.3 attests. A bounded symbolic model is, informally, a finite representation of infinite CLTLB($D$) models over the alphabet of symbolic valuations $SV(\phi)$. We restrict the analysis to ultimately periodic symbolic models, i.e., of the form $\rho = \alpha\beta\omega$. Without loss of generality, we consider models where $\alpha = \alpha' s$ and $\beta = \beta' s$ for some symbolic valuation $s$. BSP is defined with respect to a $k$-bounded model $\sigma_k : \{[\phi], \ldots, k + [\phi]\} \times V \rightarrow D$, a finite sequence $\rho'$ (with $|\rho'| = k + 1$) of symbolic valuations and a $k$-bounded satisfaction relation $|=k$ defined as follows:

$$\sigma_k,0|=k \rho' \text{ iff } \sigma_k,i|=\rho'(i) \text{ for all } 0 \leq i \leq k.$$  

The $k$-satisfiability problem of formula $\phi$ is defined as follows:
Input  A CLTLB(D) formula $\phi$, a constant $k \in \mathbb{N}$

Problem  Is there an ultimately periodic sequence of symbolic valuations $\rho = \alpha \beta^\omega$ with $|\alpha \beta| = k + 1$, $\alpha = \alpha'$'s and $\beta = \beta'$'s, such that:

- $\rho, 0 \models \phi$ and
- there is a $k$-bounded model $\sigma_k$ for which $\sigma_k, 0 \models k \alpha \beta$?

Since $k$ is fixed, the procedure for determining the satisfiability of CLTLB(D) formulae over bounded models is not complete: even if there is no accepting run of automaton $A_\phi$ when $\rho'$ as above has length $k$, there may be accepting runs for a larger $\rho'$.

Definition 4. Given a CLTLB(D) formula $\phi$, its completeness threshold $K_\phi$, if it exists, is the smallest number such that $\phi$ is satisfiable if and only if $\phi$ is $K_\phi$-satisfiable.

3.2 Avoiding explicit symbolic valuations

The next, fundamental Lemma 3 and Lemma 4 state how $k$-bounded models $\sigma_k$ are representative of ultimately periodic sequences of symbolic valuations, i.e., of symbolic models of the formula. More precisely, Lemma 4 allows for building a sequence of symbolic valuations from $\sigma_k$. Hence, the encoding described in the following Section 3.3 can consider only atomic subformulae occurring in CLTLB(D) formula $\phi$, even though the BSP for $\phi$ is defined with respect to sequences of symbolic valuations. The encoding also introduces additional constraints, to enforce periodicity of relations in $\mathcal{R}$, thus allowing us to derive an ultimately periodic symbolic model from $\sigma_k$.

To exploit this property, we adopt a special requirement on the constraint system. In fact, Lemma 3 and Lemma 4 rely on the following assumption, which guarantees the uniqueness of the symbolic valuation given an assignment to variables.

Definition 5. A constraint system $\mathcal{D} = (D, \mathcal{R})$ is value-determined if, for all $m$ and for all $v \in D^m$, there exists at most one $m$-ary relation $R \in \mathcal{R}$ s.t. $v \models_D R$.

For value-determined constraint systems we avoid the definition of set $SV(\phi)$ as we are allowed to derive symbolic models for $\phi$ through $\sigma_k$. Therefore, our approach is general and it can be used to solve CLTLB(D) for a value-determined constraint system $\mathcal{D}$, which is the case of the constraint systems presented in Section 2.4.

Lemma 3. Let $\mathcal{D} = (D, \mathcal{R})$ be a value-determined constraint system, $\phi$ be a CLTLB(D) formula and $v$ be a $\mathcal{D}$-valuation extended to terms appearing in symbolic valuations of $SV(\phi)$. Then, there is a unique symbolic valuation $sv$ such that $v \models_D sv$.

Proof. Let $sv$ be the symbolic valuation, defined from the values in $v$, such that, for any $R \in \mathcal{R}$, if $v \models_D fresh(R(\alpha_1, \ldots, \alpha_n))$ then $R(\alpha_1, \ldots, \alpha_n) \in sv$ (where fresh is the mapping introduced in Definition 2 to replace arithmetic temporal terms with fresh variables). We show that $sv$ is maximally consistent. Consistency is immediate, since if $v \models_D fresh(R(\alpha_1, \ldots, \alpha_n))$ then $v \models_D fresh(\neg R(\alpha_1, \ldots, \alpha_n))$ cannot hold. By contradiction, assume that $sv$ is not maximal, i.e., there is a relation $R' \notin sv$ such that $v \models_D fresh(R')$, and $sv \cup \{ R' \}$ is consistent. Hence, $v \models_D sv \cup \{ R' \}$. By definition,
a symbolic valuation $sv$ includes all relations among the terms of $\phi$, hence there is a relation $R'' \in sv$, with $R' \neq R''$, over the same set of terms of $R'$. Hence, in constraint system $D$ we have two different relations, $R'$ and $R''$, over the same set of terms and such that $v \models_D fresh(R')$ and $v \models_D fresh(R'')$. But this contradicts the assumption that $D$ is value-determined.  

\[ \text{Corollary 1. Let } \phi \text{ be a CLTLB}(D) \text{ formula, } v \text{ a } D\text{-valuation extended to terms of symbolic valuations and } sv \text{ a symbolic valuation in } SV(\phi). \text{ Then, for } v \models_D sv \text{ and for all relations } R \in \mathcal{R} \]

$$sv \models^{\text{sym}} R(\alpha_1, \ldots, \alpha_n) \text{ iff } v \models_D fresh(R(\alpha_1, \ldots, \alpha_n)).$$

\[ \text{Proof. Suppose that } sv \models^{\text{sym}} R(\alpha_1, \ldots, \alpha_n). \text{ By definition, } sv \models^{\text{sym}} R(\alpha_1, \ldots, \alpha_n) \text{ holds if, for every } D\text{-valuation } v' \text{ (over the set of terms in } sv) \text{ such that } v' \models_D sv, \]

$$v' \models_D fresh(R(\alpha_1, \ldots, \alpha_n)) \text{ holds. Therefore, also } v \models_D fresh(R(\alpha_1, \ldots, \alpha_n)).$$

The converse is an immediate consequence of Lemma 3. \qed

\[ \text{Lemma 4. Let } \phi \text{ be a CLTLB}(D) \text{ formula and } \sigma_k \text{ be a finite sequence of } D\text{-valuations. Then, there exists a unique locally consistent sequence } \rho \in SV(\phi)^{k+1} \text{ such that } \sigma_k, i \models \rho(i), \text{ for all } i \in [0, k]. \]

\[ \text{Proof. By Lemma 3 it follows that, for all } i \in [0, k], \text{ the assignment of variables defined by } \sigma_k \text{ is such that } \sigma_k, i \models \rho(i) \text{ and } \rho(i) \text{ is unique. By Corollary 1 values in } \sigma_k \text{ from position } i \text{ satisfy a relation } R \text{ at position } i \text{ if, and only if, } R \text{ belongs to symbolic valuation } \rho(i), \text{ i.e., } \rho(i) \models^{\text{sym}} R \text{ iff } \sigma_k, i \models fresh(R). \text{ In addition, any two adjacent symbolic valuations } \rho(i) \text{ and } \rho(i+1) \text{ are locally consistent, i.e., both } R(X^{i_1}x_1, \ldots, X^{i_n}x_n) \in \rho(i) \text{ and } R(X^{i_1-1}x_1, \ldots, X^{i_n-1}x_n) \in \rho(i+1). \text{ In fact, the evaluation in } \sigma_k \text{ of an arithmetic term } X^{i_1}x_j \text{ in position } i \text{ is the same as the evaluation of } X^{i_1-1}x_j \text{ in position } i+1. \qed \]

\[ \text{3.3 An encoding for BSP without automata} \]

We now show how to encode a CLTLB(D) formula into a quantifier-free formula in the theory EUF $\cup D$ (QF-EUD), where EUF is the theory of Equality and Uninterpreted Functions. This is the basis for reducing the BSP for CLTLB(D) to the satisfiability of QF-EUD, as proved in Section 3.4. Satisfiability of QF-EUD is decidable, provided that $D$ includes a copy of $\mathbb{N}$ with the successor relation and that EUF $\cup D$ is consistent, as in our case. The latter condition is easily verified in the case of the union of two consistent, disjoint, stably infinite theories (as is the case for EUF and arithmetic).

\[ \text{Bersani et al. (2010) describes a similar approach for the case of Integer Difference Logic (DL) constraints. It is worth noting that standard LTL can be encoded by a formula in QF-EUD with } D = (\mathbb{N}, <), \text{ rather than in Boolean logic Biere et al. (2006), resulting in a more succinct encoding.} \]

The encoding presented below represents ultimately periodic sequences of symbolic valuations $\rho$ of the form $sv_0sv_1 \ldots sv_{\text{loop}-1}(sv_{\text{loop}} \ldots sv_k)^\omega$. To do this, we use a positive integer variable $\text{loop}$ for which we require $sv_{\text{loop}-1} = sv_k$. Therefore, we look for a finite word $\rho' = sv_0sv_1 \ldots sv_{\text{loop}-1}(sv_{\text{loop}} \ldots sv_k)st_{\text{loop}}$ of length $k + 2$
representing the ultimately periodic model above. Instant $k + 1$ in the encoding is used to correctly represent the periodicity of $\rho$ by constraining atomic formulae (propositions and relations) at positions $loop$ and $k + 1$. Moreover, all subformulae of $\phi$ that hold at position $loop - 1$ must also hold in $k$.

**Encoding terms** We introduce arithmetic formula functions to encode the terms in set $terms(\phi)$. Let $\alpha$ be a term in $terms(\phi)$, then the arithmetic formula function $\alpha : \mathbb{Z} \to D$ associated with it (denoted by the same name but written in boldface), is recursively defined with respect to a finite sequence of valuations $\sigma_k$ as:

| $\alpha$ | $0 \leq i < k$ | $i = k$ |
|---|---|---|
| $x$ | $x(i) = \sigma_k(i, x)$ | $x(k) = \sigma_k(k, x)$ |
| $X\alpha'$ | $\alpha(i) = \alpha'(i + 1)$ | $\alpha(k) = \sigma_k(k + |\alpha'| + 1, x_{\alpha'})$ |

| $\alpha$ | $0 < i \leq k + 1$ | $i = 0$ |
|---|---|---|
| $Y\alpha'$ | $\alpha(i) = \alpha'(i - 1)$ | $\alpha(0) = \sigma_k(|\alpha'| - 1, x_{\alpha'})$ |

The conjunction of the above subformulae gives formula $|ArithConstraints|_k$. Implementing $|ArithConstraints|_k$ is straightforward. In fact, the assignments of values to variables are defined by the interpretation of the symbols of the QF-EUP formula. The values of variables $x$ at positions before 0 and $k$, i.e. in intervals $[\lceil \phi \rceil, -1]$ and $[k + 1, k + \lceil \phi \rceil]$, are defined by means of the values of terms $\alpha = X^i x$ and $\alpha = Y^i x$. For instance, the value of $x$ at position $0 > i \geq \lceil \phi \rceil$ is $\sigma_k(i, x)$, but it is defined by the assignment for term $\alpha = Y^i x$ at position 0.

**Encoding relations** Formula $|PropConstraints|_k$ encodes atomic subformulae $\theta$ containing relations over a.t.t.’s. Let $R$ be an $n$-ary relation of $\mathcal{R}$ that appears in $\phi$, and $\alpha_1, \ldots, \alpha_n$ be a.t.t.’s. We introduce a formula predicate $\theta : \mathbb{N} \to \{true, false\}$ — that is, a unary uninterpreted predicate denoted by the same name as the formula but written in boldface — for all $R(\alpha_1, \ldots, \alpha_n)$ in $\phi$:

| $\theta$ | $0 \leq i \leq k + 1$ |
|---|---|
| $R(\alpha_1, \ldots, \alpha_n)$ | $\theta(i) \iff R(\alpha_1(i), \ldots, \alpha_n(i))$ |
| $-R(\alpha_1, \ldots, \alpha_n)$ | $\theta(i) \iff -R(\alpha_1(i), \ldots, \alpha_n(i))$ |

**Encoding formulae** The truth value of a CLTLB formula is defined with respect to the truth value of its subformulae. We associate with each subformula $\theta$ a formula predicate $\theta : \mathbb{N} \to \{true, false\}$. When the subformula $\theta$ holds at instant $i$ then $\theta(i)$ holds. As the length of paths is fixed to $k + 1$ and all paths start from 0, formula predicates are actually subsets of \{0, \ldots, k + 1\}. Let $\theta$ be a subformula of $\phi$ and $p$ a propositional letter, formula predicate $\theta$ is recursively defined as:

| $\theta$ | $0 \leq i \leq k + 1$ |
|---|---|
| $p$ | $p(i)$ |
| $\neg p$ | $\theta(i) \iff \neg p(i)$ |
| $\psi_1 \land \psi_2$ | $\theta(i) \iff \psi_1(i) \land \psi_2(i)$ |
| $\psi_1 \lor \psi_2$ | $\theta(i) \iff \psi_1(i) \lor \psi_2(i)$ |

15
The conjunction of the formulae above is also part of formula $|PropConstraints|_k$. The temporal behavior of future and past operators is encoded in formula $|TempConstraints|_k$ by using their traditional fixpoint characterizations. More precisely, $|TempConstraints|_k$ is the conjunction of the following formulae, for each temporal subformula $\theta$:

| $\theta$ | $0 \leq i \leq k$ |
|----------|------------------|
| $X\psi$  | $\theta(i) \Leftrightarrow \psi(i + 1)$ |
| $\psi_1 U \psi_2$ | $\theta(i) \Leftrightarrow (\psi_2(i) \lor (\psi_1(i) \land \theta(i + 1)))$ |
| $\psi_1 R \psi_2$ | $\theta(i) \Leftrightarrow (\psi_2(i) \land (\psi_1(i) \lor \theta(i + 1)))$ |

| $\theta$ | $0 < i \leq k + 1$ | $i = 0$ |
|----------|------------------|---------|
| $Y\psi$  | $Y\psi(i) \Leftrightarrow \psi(i - 1)$ | $false$ |
| $\psi_1 S \psi_2$ | $\theta(i) \Leftrightarrow (\psi_2(i) \lor (\psi_1(i) \land \theta(i - 1)))$ |
| $\psi_1 T \psi_2$ | $\theta(i) \Leftrightarrow (\psi_2(i) \land (\psi_1(i) \lor \theta(i - 1)))$ |

$\psi_1 \psi_2$ denotes the conjunction of the following formulae, for each temporal subformula $\theta$:

$$\theta = R(\alpha_1, \ldots, \alpha_n) \land \left( \bigwedge_{R \in \mathcal{R}, \alpha_1, \ldots, \alpha_n \in terms(\phi)} \theta(loop - 1) = \theta(k). \right)$$

Last state constraints (captured by formula $|LastStateConstraints|_k$) define the equivalence between the truth values of the subformulae of $\phi$ at position $k + 1$ and those at the position indicated by the $loop$ variable, since the former position is representative of the latter along periodic paths. These constraints have a similar structure as those in the Boolean encoding of [Biere et al. (2006)]. For brevity, we consider only the case for infinite periodic words, as the case for finite words can be easily achieved. Hence, last state constraints are introduced through the following formula (where $sub(\phi)$ indicates the set of subformulae of $\phi$) by adding only one constraint for each subformula $\theta$ of $\phi$:

$$\bigwedge_{\theta \in sub(\phi)} \theta(k + 1) \iff \theta(loop).$$

Eventualities for $U$ and $R$ To correctly define the semantics of $U$ and $R$, their eventualities have to be accounted for. Briefly, if $\psi_1 U \psi_2$ holds at $i$, then $\psi_2$ eventually holds in some $j \geq i$; if $\psi_1 R \psi_2$ does not hold at $i$, then $\psi_2$ eventually does not hold in some $j \geq i$. Along finite paths of length $k$, eventualities must hold between $0$ and $k$. Otherwise, if there is a loop, an eventuality may hold within the loop. The Boolean encoding of [Biere et al. (2006)] introduces $k$ propositional variables for each subformula $\theta$ of $\phi$ of the form $\psi_1 U \psi_2$ or $\psi_1 R \psi_2$ (one for each $1 \leq i \leq k$), which represent the
eventuality of $\psi_2$ implicit in the formula. Instead, in the QF-EUD encoding, only one variable $j_{\psi_2} \in D$ is introduced for each $\psi_2$ occurring in a subformula $\psi_1 U \psi_2$ or $\psi_1 R \psi_2$.

\[
\begin{array}{c|c}
\psi_1 U \psi_2 & \theta(k) \Rightarrow \text{loop} \leq j_{\psi_2} \leq k \land \psi_2(j_{\psi_2}) \\
\psi_1 R \psi_2 & -\theta(k) \Rightarrow \text{loop} \leq j_{\psi_2} \leq k \land -\psi_2(j_{\psi_2})
\end{array}
\]

The conjunction of the constraints above for all subformulae $\theta$ of $\phi$ constitutes the formula $|\text{Eventually}|_k$.

The complete encoding $|\phi|_k$ of $\phi$ consists of the logical conjunction of all above components, together with $\phi$ evaluated at the first instant of time.

### 3.4 Correctness of the BSP encoding

To prove the correctness of the encoding defined in Section 3.3 we first introduce two properties, which reduce CLTLB($D, R_0$) to CLTLB($D$) without $Y$ operators. This allows us to base our proof on the automata-based construction for CLTLB($D$) of Demri and D’Souza (2007). In particular, the two reductions are essential to take advantage of Proposition 2 and Lemma 2 of Section 2 to define a decision procedure for the bounded satisfiability problem of Section 3.1. The properties are almost obvious, hence we only provide the intuition behind their proof (see Bersani et al. (2012) for full details).

**Property 2.** CLTLB($D, R_0$) formulae can be equivalently rewritten into CLTLB($D$) formulae.

According to the definition given in Section 2.2 CLTLB($D$) is the language CLTLB where atomic formulae belong to the language of constraints in $D$, which may contain also 0-ary relations. In this case, atomic formulae are propositions $p \in R_0$ or relations $R(\alpha_1, \ldots, \alpha_n)$. Any positive occurrence of an atomic proposition $p \in R_0$ in a CLTLB formula can be replaced by an equality relation of the form $x_p = 1$. Then, a formula of CLTLB($D, R_0$) can be easily rewritten into a formula of CLTLB($D$) preserving the equivalence between them (modulo the rewriting of propositions in $R_0$). We define a rewriting function $np$ over formulae such that $(\pi', \sigma'), 0 \models p$ if, and only if, $(\pi, \sigma), 0 \models np(\phi) \land \psi$ where $\sigma$ is the same as $\sigma'$ except for new fresh variables $x_p$ representing atomic propositions, and $\psi$ is a formula restricting the values of variables $x_p$ to $\{0, 1\}$.

For instance, let $\phi$ be the formula $G(p \Rightarrow F(Xx < y \land xq))$, where the “eventually” ($F$) and “globally” ($G$) operators are defined as usual. The formula obtained by means of rewriting $np$ is

$$G(x_p = 1 \Rightarrow F(Xx < y \land xq = 1)) \land G \left( \begin{array}{l} (x_p = 1 \lor x_p = 0) \\
\land \\
(x_q = 1 \lor x_q = 0) \end{array} \right).$$

Note that formula $np(\phi)$ does not contain any propositional letters, so in a model $(\pi, \sigma)$ component $\pi$ associates with each instant the empty set. From now on we will consider only CLTLB($D$) formulae without propositional letters; hence, given a propositional letter-free formula $\phi$, we will write $\sigma, 0 \models \phi$ instead of $(\pi, \sigma), 0 \models \phi$. 

17
Property 3. CLTLB(\(D\)) formulae can be equivalently rewritten into CLTLB(\(D\)) formulae without \(Y\) operators.

Let \(sl : \text{CLTLB}(\mathcal{D}) \rightarrow \text{CLTLB}(\mathcal{D})\) be the following mapping, which transforms (by "shifting to the left") every formula \(\phi\) into an equisatisfiable formula that does not contain any occurrence of the \(Y\) operator. Formula \(sl(\phi)\) is identical to \(\phi\) except that all a.t.t.’s of the form \(X^i x\) in \(\phi\) are replaced by \(X^{i-\lfloor \phi \rfloor} x\), while all a.t.t.’s of the form \(Y^i x\) are replaced by \(X^{-i-\lfloor \phi \rfloor} x\). The latter replacement avoids negative indexes (since if \(\phi\) contains a.t.t.’s of the form \(Y^i x\), then \(\lfloor \phi \rfloor < 0\)). The \(sl\) function can be naturally extended to symbolic valuations (i.e, sets of atomic constraints) and sequences \(\rho\) thereof.

As a consequence, given a CLTLB(\(D\)) formula \(\phi\), it is easy to see that \(Y\) does not occur in \(sl(\phi)\). The equisatisfiability of formulae \(\phi\) and \(sl(\phi)\) is guaranteed by moving the origin of \(\phi\) by \(-\lfloor \phi \rfloor\) instants in the past. Since only \(X\) occurs in \(sl(\phi)\), then models for CLTLB(\(D\)) formulae without \(Y\) are now sequences of \(\mathcal{D}\)-valuations \(\sigma : \mathbb{N} \times V \rightarrow D\).

Proposition 4. Let \(\phi\) be a CLTLB(\(D\)) formula, then \(\sigma, 0 \models \phi\) iff \(\sigma, \lfloor \phi \rfloor \models sl(\phi)\).

Corollary 2. Let \(\rho \in SV(\phi)^{\omega}\) be a sequence of symbolic valuations. Then,

\[
\sigma, 0 \models \rho \iff \sigma, \lfloor \phi \rfloor \models sl(\rho), \rho, 0 \models \text{sym} \iff sl(\rho), 0 \models \text{sym} \models sl(\phi).
\]

We now have all necessary elements to prove the correctness of our encoding. We first provide the following three equivalences, which are proved by showing the implications depicted in Figure 1, where \(A_s \times A_\ell\) is the automaton recognizing symbolic models of \(sl(\phi)\):

1. Satisfiability of \(|\phi|_k\) is equivalent to the existence of ultimately periodic runs of automaton \(A_s \times A_\ell\).
2. \(k\)-satisfiability is equivalent to the existence of ultimately periodic runs of automaton \(A_s \times A_\ell\).
3. \(k\)-satisfiability is equivalent to the satisfiability of \(|\phi|_k\).

Then we draw, by Proposition 5, the connection between \(k\)-satisfiability and satisfiability for formulae over constraint systems satisfying the completion property. In Section 4, thanks to Proposition 6, we extend the result to constraint system IPC*, which does not have the completion property.

Before tackling the theorems of Figure 1, we provide the definition of models for QF-EUD formulae \(|\phi|_k\) built according to the encoding of Section 3.3. More precisely, a model \(M\) of \(|\phi|_k\) is a pair \((\mathcal{D}, \mathcal{I})\) where \(\mathcal{D}\) is the domain of interpretation of \(\mathcal{D}\), and \(\mathcal{I}\) maps

- each function symbol \(\alpha\) onto a function associating, for each position of time, an element in domain \(D, \mathcal{I}(\alpha) : \mathbb{N} \rightarrow D\);
each predicate symbol \( \theta \) onto a function associating, for each position of time, an element in \{true, false\}. Note that mapping \( I \) trivially induces a finite sequence of \( D \)-valuations \( \sigma_k : \{\lfloor \phi \rfloor, \ldots, k + \lceil \phi \rceil\} \to D \).

We start by showing that the existence of ultimately periodic runs of automaton \( A_s \times A_{\ell} \) implies the satisfiability of \( |\phi|_k \).

**Theorem 1.** Let \( \phi \in \text{CLTLB}(D) \) with \( N \) definable in \( D \) together with the successor relation. If there exists an ultimately periodic run \( \rho = \alpha \beta \omega \) of \( A_s \times A_{\ell} \) accepting symbolic models of \( \text{sl}(\phi) \), then \( |\phi|_k \) is satisfiable with respect to \( k \in \mathbb{N} \).

In the following proof, we use the generalized Büchi automaton obtained by the standard construction of Vardi and Wolper (1986), in the version of Demri and D’Souza (2007). Let \( \phi' \) be a CLTLB(D) formula (without the Y modality over terms). The closure of \( \phi' \), denoted \( \text{cl}(\phi') \), is the smallest negation-closed set containing all subformulae of \( \phi \). An atom \( \Gamma \subseteq \text{cl}(\phi') \) is a subset of formulae of \( \text{cl}(\phi') \) that is maximally consistent, i.e., such that, for each subformula \( \xi \) of \( \phi' \), either \( \xi \in \Gamma \) or \( \neg \xi \in \Gamma \). A pair \((\Gamma_1, \Gamma_2)\) of atoms is one-step temporally consistent when:

- for every \( X \psi \in \text{cl}(\phi') \), then \( X \psi \in \Gamma_1 \) iff \( \psi \in \Gamma_2 \),
- for every \( Y \psi \in \text{cl}(\phi') \), then \( Y \psi \in \Gamma_2 \) iff \( \psi \in \Gamma_1 \),
- if \( \psi_1 U \psi_2 \in \Gamma_1 \), then \( \psi_2 \in \Gamma_1 \) or both \( \psi_1 \in \Gamma_1 \) and \( \psi_1 U \psi_2 \in \Gamma_2 \),
- if \( \psi_1 S \psi_2 \in \Gamma_2 \), then \( \psi_2 \in \Gamma_2 \) or both \( \psi_1 \in \Gamma_2 \) and \( \psi_1 S \psi_2 \in \Gamma_1 \).

The automaton \( A_s = (SV(\phi'), Q, Q_0, \eta, F) \) is then defined as follows:

- \( Q \) is the set of atoms;
- \( Q_0 = \{ \Gamma \in Q : \phi' \in \Gamma, Y \psi \notin \Gamma \) for all \( \psi \in \text{cl}(\phi'), \psi_1 S \psi_2 \in \Gamma \) iff \( \psi_2 \in \Gamma \} \);
- \( \Gamma_1 \xrightarrow{sv} \Gamma_2 \in \eta \) iff
  - \( sv \mid_{SV} \Gamma_1 \)
  - \((\Gamma_1, \Gamma_2)\) is one-step consistent;
Consider the sequence \( \rho \) be uniquely defined by considering at each position \( i \) of symbolic valuations is consistent and all the a.t.t.'s in the encoding of \( \rho \) for each position in \( D \) defined by using the same Boolean closure in \( | \theta \rangle \in \text{loop} \). In particular, \( \rho \) the interpretation of subformulae at position \( k \) translated by means of subformulae occurring in every \( \rho \). The truth value of subformulae \( \psi \), for uninterpreted predicate and functions formulae: given a symbolic model for \( A_s \times A_t \) and for each subformula \( \phi \) occurring in atoms of \( \gamma \), we obtain an accepting run \( s \), which associates, for each variable \( i \) of \( \rho \), the interpretation of subformulae at position \( i \). Note that by taking truth values of subformulae \( \psi \) on the alphabet of \( A_s \times A_t \) (for simplicity, and without loss of generality, we assume that \( sv_{\text{loop}} = sv_k \)). \( \rho \) is recognized by a periodic run of \( A_s \times A_t \) of the form:

\[
\rho = sv_0 \ldots sv_{\text{loop}}(sv_{\text{loop}})\omega
\]

such that \( \rho \in \mathcal{L}(A_s \times A_t) \) (for simplicity, and without loss of generality, we assume that \( sv_{\text{loop}} = sv_k \)). \( \rho \) is recognized by a periodic run of \( A_s \times A_t \) of the form:

\[
v = (\Gamma_0, sv_0) \ldots (\Gamma_{\text{loop}}-1, sv_{\text{loop}}-1)(\Gamma_{\text{loop}}, sv_{\text{loop}}) \ldots (\Gamma_k, sv_k))\omega.
\]

For each subformula \( \psi \) occurring in \( \phi \), subrun \( (\Gamma_{\text{loop}}-1, sv_{\text{loop}}-1)(\Gamma_{\text{loop}}, sv_{\text{loop}}) \ldots (\Gamma_k, sv_k) \) visits control states of the set \( F_i \), thus witnessing the acceptance condition of \( A_s \). From \( v \) we build run \( \gamma \) of \( A_s \):

\[
\gamma = \Gamma_0 \ldots \Gamma_{\text{loop}}-1(\Gamma_{\text{loop}} \ldots \Gamma_k)\omega.
\]

In particular, \( \rho \) is defined by the projection on the alphabet of \( SV(sl(\phi)) \) of the subformulae occurring in every \( \Gamma_i \), for \( 0 \leq i \leq k \). Sequence \( \rho \) and its accepting run \( \gamma \) can be translated by means of \( sl^{-1} \) to obtain a symbolic model for \( \phi \). In particular, because \( 0 |\equiv sl(\phi) \) then we obtain, by Corollary 2, \( sl^{-1}(\rho), 0 |\equiv \phi \). Similarly, by shifting all formulae in atoms of \( \gamma \), we obtain an accepting run \( sl^{-1}(\gamma) \) for \( \phi \). The model for \( |\phi|_k \) is given by the truth value of all the subformulae in each \( sl^{-1}(\Gamma_i) \) and the values of variables occurring in \( \phi \) can be defined as explained later. In particular, we need to complete interpretation \( I \) for uninterpreted predicate and functions formulae: given a position \( 0 \leq i \leq k \), for all subformulae \( \theta \in cl(\phi) \) we define

- \( I(\theta)(i) = \text{true} \iff \theta \in sl^{-1}(\Gamma_i) \),
- \( I(\theta)(i) = \text{false} \iff \neg\theta \in sl^{-1}(\Gamma_i) \).

The truth value of subformulae \( \psi \chi \) and \( \psi \gamma \) is derived by duality. To complete the interpretation of subformulae at position \( k \) we can use values from position \( \text{loop} \): \( I(\theta)(k + 1) = I(\theta) \langle \text{loop} \rangle \). Note that by taking truth values of subformulae \( \theta \in cl(\phi) \) from atoms \( sl^{-1}(\Gamma_i) \), \( \text{propConstraints} |_k \) are trivially satisfied (atoms are defined by using the same Boolean closure in \( \text{propConstraints} |_k \)). The sequence \( \rho \) of symbolic valuations is consistent and all the a.t.t.'s in the encoding of \( |\phi|_k \) can be uniquely defined by considering at each position \( i \) a symbolic valuation \( sl^{-1}(sv_i) \). Consider the sequence \( \rho' = sv_0 \ldots sv_{\text{loop}}-1(sv_{\text{loop}}) sv_{\text{loop}} \). Following Demri and D’Souza [2007] Lemma 5.2), we can build an edge-respecting assignment of values in \( D \) for the finite graph \( G_{sl^{-1}(\rho')} \), which associates, for each for each variable \( x \in V \) and for each position \( |\phi| \leq i \leq k + 1 \), a value \( \sigma_k(i, x) \). We exploit assignment

\footnote{\text{For reasons of clarity, we avoid some details of product automaton } A_s \times A_t, \text{ which are however inessential in the proof.}}
\(\sigma_k(i, x)\) to define \(I(\alpha)\), with \(\alpha \in \text{terms}(\phi)\), in the following way (where \(x_{\alpha}\) is the variable in \(\alpha\)):

\[
I(\alpha)(i) = \sigma_k(i + |\alpha|, x_{\alpha})
\]

for all \(0 \leq i \leq k + 1\). Then, formulae \(|ArithConstraints|_k\) are satisfied. Since run \(v\) is ultimately periodic, then control state \((\Gamma_{loop}, sv_{loop})\) is visited at position \(k + 1\). It witnesses the satisfaction of \(|LastStateConstraints|_k\) formulae, which prescribe that \(\theta_{k+1} \iff \theta_{loop}\) for all \(\theta \in cl(\phi)\). Finally, let us consider \(|Eventually|_k\) formulae. If subformula \(\varphi = \psi U \zeta\) belongs to atom \(\Gamma_k\), then there exists a position \(j \geq k\) such that \(\zeta\) holds in \(j\). Since the model is periodic, then \(k \leq j \leq k + |\beta|\), i.e., \(j_\zeta = j - |\beta|\) is a position such that \(\text{loop} \leq j_\zeta \leq k\). Moreover, if \(\neg (\psi R \zeta) = \neg \psi U \neg \zeta\) belongs to \(\Gamma_k\) then there exists a position \(j \geq k\) such that \(\neg \zeta\) holds in \(j\). As in the previous case \(\text{loop} \leq j_\zeta \leq k\). Hence, the \(|Eventually|_k\) formulae are satisfied. The initial atom \(\Gamma_0\) is such that \(Y \varphi \not\in \Gamma_0\) and if \(\psi S \zeta \in \Gamma_0\) then \(\zeta \in \Gamma_0\), which witnesses the encoding of subformulae \(Y \psi\) and \(\psi S \zeta\) at \(0\), i.e., \(\theta_0 \iff \bot\) and \(\theta_0 \iff \zeta_0\), respectively.

We now prove the second implication, which draws the connection between the encoding and the \(k\)-satisfiability problem.

**Theorem 2**. Let \(\phi \in \text{CLTLB}(D)\) with \(\mathbb{N}\) definable in \(D\) together with the successor relation. If \(|\phi|_k\) is satisfiable, then formula \(\phi\) is \(k\)-satisfiable with respect to \(k \in \mathbb{N}\).

**Proof**. We prove the theorem by showing that formula \(|\phi|_k\) defines ultimately periodic symbolic models \(\rho = \alpha \beta^\omega\) for formula \(\phi\) such that \(\sigma_k, 0 \models_k \alpha \beta\) and \(\rho, 0 \models \phi\). Note that the encoding of \(|\phi|_k\) defines precisely the truth value of all subformulae \(\theta\) of \(\phi\) in instants \(i \in [0, k]\). Then, if \(|\phi|_k\) is satisfiable, given an \(i \in [0, k]\), the set of all subformulae

\[
\Gamma_i = \{ \varphi \in cl(\phi) \mid \text{if } \theta(i) \text{ holds then } \varphi = \theta, \text{ else } \varphi = \neg \theta \}
\]

is a maximal consistent set of subformulae of \(\phi\). We have \textit{loop} \(\in [1, k]\). The sequence of sets \(\Gamma_i\) for \(0 \leq i \leq k\) is an ultimately periodic sequence of maximal consistent sets due to formulae \(|LastStateConstraints|_k\) and \(|LoopConstraints|_k\). We write \(\Gamma|A\) to denote the projection of \(D\)-constraints in \(\Gamma\) on symbols of the set \(A\); e.g., if \(A = \{ R_1, R_2 \}\) then \(\{ R_1(x, y), R_2(Xx, Y y), \theta_1, \theta_2 \}|_A = \{ R_1(x, y), R_2(Xx, Y y) \}\). The sequence of atoms is

\[
\gamma = \Gamma_0 \ldots \Gamma_{\text{loop} - 1} (\Gamma_{\text{loop}} \ldots , \Gamma_k)^\omega
\]

and such that \(\Gamma_{\text{loop} - 1}|_R\) is equal to the set of relations of \(\Gamma_k|_R\) by \(|LoopConstraints|_k\) formulae. Moreover, by \(|LastStateConstraints|_k\) we have \(\Gamma_{k+1} = \Gamma_{\text{loop}}\).

By Lemma [4] from the bounded sequence \(\sigma_k\) of \(D\)-valuations induced by \(I\), we have a unique locally consistent finite sequence of symbolic valuations \(\alpha \beta\) such that \(\sigma_k, 0 \models_k \alpha \beta\). Formula \(|LoopConstraints|_k\) witnesses ultimately periodic sequences of symbolic valuations \(\rho\) because it is defined over the set of relations in \(R\) and all terms of the set \(\text{terms}(\phi)\):

\[
\rho = \alpha \beta^\omega = sv_0 \ldots sv_{\text{loop} - 1}(sv_{\text{loop}} \ldots sv_k)^\omega
\]

21
ultimately periodic run to some $k$.

We call $\rho^i$ the suffix of $\rho$ that starts from position $i \geq 0$. By structural induction on $\phi$ one can prove that for all $0 \leq i \leq k + 1$, for all subformulæ $\theta$ of $\phi$, $\theta(i)$ holds (i.e., $\theta \in \Gamma_1$) if, and only if,

- $\rho^i, 0 \models \theta$ for $\theta$ of the form $R, X, U, R$;
- $(sv_0 \ldots sv_i), i \models \theta$ for $\theta$ of the form $Y, S, T$.

Then, since by hypothesis $\phi(0)$ holds, we have that $\rho, 0 \models \phi$.

The base case is the unique fundamental part of the proof because the inductive step over temporal modalities is rather standard. Let us consider a relation formula $\theta$ of the form $R(\alpha_1, \ldots, \alpha_n)$ where, for all $1 \leq j \leq n$, $\alpha_j \in \text{terms}(\phi)$. We have to show that $\theta(i)$ holds if, and only if, $sv_i \models \theta$. We have that $\theta(i)$ holds if, and only if, $\alpha_{k,i} \models \theta$; since, by Lemma 4, $\alpha_k, i \models \theta$ if, and only if, the symbolic valuation $sv_i$ induced by $\alpha_k$ at $i$ includes $\theta$, we have by definition $sv_i \models \theta$.

We omit the inductive step, which is standard and is reported in [Biere et al., 2006] and [Pradella et al., 2013], since we use the same operators with the same encodings.

Finally, the next theorem draws a link between $k$-satisfiability and the existence of an ultimately periodic run in automaton $A_s \times A_\ell$.

**Theorem 3.** Let $\phi \in \text{CLTLB}(D)$ with $\mathbb{N}$ definable in $D$ together with the successor relation. If formula $\phi$ is $k$-satisfiable with respect to $k \in \mathbb{N}$, then there exists an ultimately periodic run $\rho = \alpha \beta^\omega$ of $A_s \times A_\ell$, with $|\alpha \beta| = k + 1$, accepting symbolic models of $sl(\phi)$.

**Proof.** By definition, if $\phi$ is $k$-satisfiable so is $sl(\phi)$, and there is an ultimately periodic symbolic model $\rho = \alpha \beta^\omega$ such that $\rho, 0 \models sl(\phi)$. By Lemma 4, $\rho$ is locally consistent because there exists a $k$-bounded model $\sigma_k$ such that $\sigma_k \models \alpha \beta$. Therefore, $\rho \in \mathcal{L}(A_s \times A_\ell)$.

As explained in Section 2, each automaton involved in the definition of $A_\phi$ has the function of “filtering” sequences of symbolic valuations so that 1) they are locally consistent, 2) they satisfy an LTL property and 3) they admit a (arithmetic) model. As mentioned in Section 2, for constraint systems that have the completion property local consistency is a sufficient and necessary condition for admitting a model. For these constraint systems $A_\phi$ is exactly automaton $A_s \times A_\ell$, and from Proposition 2 and Theorem 3 we obtain the following result.

**Proposition 5.** Let $\phi \in \text{CLTLB}(D)$ with $\mathbb{N}$ definable in $D$ together with the successor relation and satisfying the completion property. Formula $\phi$ is $k$-satisfiable with respect to some $k \in \mathbb{N}$ if, and only if, there exists a model $\sigma$ such that $\sigma, 0 \models \phi$.

**Proof.** Suppose formula $\phi$ is $k$-satisfiable. Then, by Theorem 3 there is a symbolic model $\rho = \alpha \beta^\omega$ such that $\rho, 0 \models sl(\phi)$. By Proposition 2 $\rho$ admits a model $\sigma'$, i.e., such that $\sigma', 0 \models sl(\phi)$. By Corollary 2 we have $\sigma', -[\phi] \models \phi$, so the desired $\sigma$ is simply $\sigma'$ translated of $[\phi]$.
 Conversely, if formula $\phi$ is satisfiable, then automaton $A_{sl(\phi)}$ recognizes a nonempty language in $SV(sl(\phi))^\omega$. Hence, there is an ultimately periodic, locally consistent, sequence of symbolic valuations $\rho = \alpha\beta^\omega$, with $|\alpha\beta| = k + 1$, which is accepted by automaton $A_{sl(\phi)}$. Then, the model $\sigma_k$ that shows the $k$-satisfiability of $\phi$ is built considering prefix $\alpha\beta$, by defining an edge-respecting labeling of graph $G_{\alpha\beta}$.

When constraint systems do not have the completion property, locally consistent symbolic models $\rho$ recognized by automaton $A_s \times A_t$ may not admit arithmetical models $\sigma$ such that $\sigma \models \rho$. However, as mentioned in Section 2.4.1, for some constraint systems $D$, it is possible to define a condition $C$ over symbolic models such that $\rho \in \ell(A_s \times A_t)$ satisfies $C$ if, and only if $\rho$ admits a model. We tackle this issue in the next section.

4 Bounded Satisfiability of CLTLB(IPC*)

When $D$ is IPC*, Proposition 5 does not apply since, by Lemma 1, IPC* does not have the completion property. However, as shown by Lemma 2, ultimately periodic symbolic models of CLTLB(IPC*) formulae admit arithmetic model if, and only if, they obey the condition captured by Property 1. In this section, we define a simplified condition of (non) existence of arithmetical models for ultimately periodic symbolic models of CLTLB(IPC*) formulae, and we show its equivalence with Property 1. Then, we provide a bounded encoding through QF-EUD formulae (where $D$ embeds $\mathbb{N}$ and the successor function) for the new condition, and we define a specialized version of Proposition 5 for $D = IPC^*$. Finally, we introduce simplifications to the encoding that can be applied in special cases.

Let $\rho$ be a symbolic model for CLTLB(IPC*) formula $\phi$. To devise the simplified condition equivalent to Property 1, we provide a specialized version of graph $G_\rho$ where points are identified by their relative position within symbolic valuations. We introduce the notion of point $p = (x, j, h)$ in $\rho$ which we use to identify a variable or a constant $x \in V \cup \text{const}(\phi)$ at position $h$ within symbolic valuation $\rho(j)$; i.e., we refer to variable $x$, or constant $c$, at position $j + h$ of the symbolic model $\rho$. Given a point $p = (x, j, h)$ of $\rho$, we denote with $\text{var}(p)$ the variable $x$, with $\text{sv}(p)$ the symbolic valuation $j$ (with $\text{sv}(p) \geq 0$), and with $\text{shift}(p)$ the position of $x$ within the $j$-th symbolic valuation (with $\text{shift}(p) \in [[\phi], [\phi]]$); also, $x(j + h)$ is the value of variable $x$ in position $h$ of the $j$-th symbolic valuation of $\rho$. Given a symbolic model $\rho$, we indicate by $P_\rho$ the set of points of $\rho$.

Different triples can refer to equivalent points. For example, variable $x$ in position 2 of symbolic valuation 4 (i.e., $(x, 4, 2)$) is the same as $x$ in position 1 of adjacent symbolic valuation 5 (i.e., $(x, 5, 1)$), and also of $x$ in position 0 of symbolic valuation 6 (i.e., $(x, 6, 0)$). Figures 2 and 3 show examples of equivalent points. Hence, we need to define an equivalence relation between triples, called local equivalence.

Definition 6. For all points $p_1 = (x, j, h)$, $p_2 = (x, i, m)$ in $P_\rho$, we say that $p_1$ is locally equivalent to $p_2$ if $j + h = i + m$, with $i, j \geq 0$ and $h, m \in [[\phi], [\phi]]$.

Definition 7. We define the relation $p_1 \preceq p_2$ in $P_\rho$. Given $p_1 = (x, j, h)$ and $p_2 = (y, i, m)$ of $P_\rho$, it is $p_1 \preceq p_2$ if:

23
1. \( i + m - (j + h) < -[\phi] + [\phi] + 1 \).
2. \( j + h \leq i + m \)
3. \( x(j + h) \leq y(i + m) \)

Similarly, relations \(<, \geq, \succ, \preceq\) \(\subseteq P_p \times P_p\) are defined as above by replacing \(\leq\) with, respectively, \(<, \geq, \succ, \preceq\) in Condition 3.

By Condition 1 of Definition 7, for each relation \(\sim\in \{\preceq\}\), \(p_1 \sim p_2\) may hold only if the distance between \(p_1\) and \(p_2\) is smaller than the size \(-[\phi] + [\phi] + 1\) of a symbolic valuation, i.e., \(p_1\) and \(p_2\) are “local”, in the sense that they belong either to the same symbolic valuation (i.e., \(j = i\)) or to the common part of “partially overlapping” symbolic valuations (see Figures 2 and 3 for examples of partially overlapping symbolic valuations). By Condition 2, each relation \(\sim\) is a positional precedence, i.e., if \(p_1 \sim p_2\) then \(p_2\) cannot positionally precede \(p_1\). Condition 3 is well defined on symbolic valuations, since it corresponds to having, in graph \(G_p\), an arc between \(p_1\) and \(p_2\) that is labeled with \(\sim\). The reflexive relations \(<, \geq\) have an antisymmetric property, in the sense that if \(p_1 \lesssim p_2\) and \(p_2 \lesssim p_1\), then \(p_1 \simeq p_2\) and \(p_2 \simeq p_1\) (analogously for \(\geq\): if \(p_1 = (x, j, h)\) and \(p_2 = (y, i, m)\), then \(p_1\) and \(p_2\) are at the same position \(j + h = i + m\) and have the same value \(x(j + h) = y(i + m)\).

Notice that the relations \(\sim\) are not transitive, because of Condition 1: each relation \(\sim\) is only “locally” transitive, in the sense that if \(p_1 \sim p_2\) and \(p_2 \sim p_3\), then \(p_1 \sim p_3\) if, and only if, Condition 1 holds for \(p_1\) and \(p_3\) (i.e., when also \(p_1, p_3\) are “local”, which in general may not be the case).

**Definition 8.** We say that there is a local forward (resp. local backward) path from point \(p_1\) to point \(p_2\) if \(p_1 \preceq p_2\) (resp., \(p_1 \succeq p_2\)); the path is called strict if \(p_1 \prec p_2\) (resp., \(p_1 \succ p_2\)).

Obviously, given two points \(p_1 = (x, j, h)\) and \(p_2 = (y, i, m)\) of \(P_p\) such that \(|i+m-(j+h)| < -[\phi] + [\phi] + 1\), it must be at least one of \(p_1 \preceq p_2, p_2 \preceq p_1, p_1 \succeq p_2, p_2 \succeq p_1\); if it is both \(p_1 \preceq p_2\) and \(p_1 \succeq p_2\), then \(p_1 \simeq p_2\), hence \(x(j + h) = y(i + m)\).

It is immediate to notice that the local equivalence is a congruence for all relations, e.g., if \(p_1\) is locally equivalent to \(p'_1\) and \(p_2\) is locally equivalent to \(p'_2\) then \(p_1 \preceq p_2\) iff \(p'_1 \preceq p'_2\). Figures 2 and 3 depict examples of this fact.

We now extend the relations of Definition 8 to cope with non-overlapping symbolic valuations.

**Definition 9.** Relation \(\preceq\) \(\subseteq P_p \times P_p\), for every \(\sim\in \{\preceq\}\), denotes the transitive closure of \(\sim\). Relations \(\preceq\), \(\succeq\) \(\subseteq P_p \times P_p\), are defined as follows, for all \(p_1, p_2 \in P_p\):

- \(p_1 \preceq p_2\) if there exist \(p', p'' \in P_p\) such that \(p_1 \preceq p' \prec p'' \preceq p_2\);
- \(p_1 \succeq p_2\) if there exist \(p', p'' \in P_p\) such that \(p_1 \succeq p' \succ p'' \succeq p_2\).

**Remark 1.** If \(p_1 = (x, j, h), p_2 = (y, i, m)\) and \(p_1 \preceq p_2\), then it is \(x(j + h) \leq y(i + m)\). The other cases of \(\preceq\) are similar. If \(\sim\) is, respectively, \(<, \preceq, \succ, \succeq\), then relation between \(x(j + h)\) and \(y(i + m)\) is, respectively, \(<, =, \succ, \preceq\), if it is \(p_1 \preceq p_2\).
Figure 2: Adjacent and overlapping symbolic valuations $\rho(i)$ (solid line) and $\rho(i - 2)$ (dotted line) of length 3 (with $-\lfloor \varphi \rfloor = \lceil \varphi \rceil = 1$), with $p_1 = (y, i, -1)$ and $p'_1 = (y, i - 2, 1)$ being locally equivalent. Both $p_1 \sim p_2$ and $p'_1 \sim p_2$ hold.

Figure 3: Adjacent and overlapping symbolic valuations $\rho(i)$ (solid line) and $\rho(i + 1)$ (dotted line) of length 3 (with $-\lfloor \varphi \rfloor = \lceil \varphi \rceil = 1$), with points $p_2 = (x, i, 1)$ and $p'_2 = (x, i + 1, 0)$ being locally equivalent. Both $p_1 \sim p_2$ and $p'_1 \sim p'_2$ hold.

but not $p_1 \prec p_2$, then along the path from $p_1$ to $p_2$ there are only arcs labeled with $\approx$, i.e. $p_1 \succcurlyeq p_2$, so $x(j + h) = y(i + m)$. As a consequence, if it is $p_1 \succeq p_2$, but not $p_1 \succcurlyeq p_2$, then it is also $p_1 \succ p_2$. The dual properties hold for $\succcurlyeq$ and $\succ$.

Let $\rho = \alpha \beta^\omega \in SV(\phi)^\omega$ be an ultimately periodic symbolic model of $\phi$. We need to introduce another notion of equivalence, which is useful for capturing properties of points of symbolic valuations in $\beta^\omega$, though it is defined in general. More precisely, we consider two points $p, p' \in P_\rho$ as equivalent when they correspond to the same variable, in the same position of the symbolic valuation, but in symbolic valuations that are $i | \beta |$ positions apart, for some $i \geq 0$. In fact, points in $\beta^\omega$ that are equivalent according to the definition below have the same properties concerning forward and backward paths.

**Definition 10.** Two points $p, p' \in P_\rho$ are equivalent, written $p \equiv p'$, when $\text{var}(p) = \text{var}(p')$, $\text{sv}(p') = \text{sv}(p) + i | \beta |$ and $\text{shift}(p) = \text{shift}(p')$, for some $i \in \mathbb{Z}$.

The main result of the section is Formula (1) on page 29 which is based on a number of intermediate results that are presented in the following. To test for the condition for the existence of arithmetic models of symbolic model $\rho = \alpha \beta^\omega$, one must represent infinite (possibly strict) forward and backward paths along $\rho$. To this end, we devise a condition for the existence of infinite paths, resulting from iterating suffix $\beta$ infinitely many times. Without loss of generality, in the following we consider ultimately periodic models $\rho = \alpha \beta^\omega$ in which $\alpha = \alpha'$ and $\beta = \beta'$, i.e., in which the last symbolic valuation of prefix $\alpha$ is the same as the last symbolic valuation of repeated suffix $\beta$. We indicate by $k + 1$ the length of $\alpha \beta$, and we number the symbolic valuations in $\alpha \beta$ starting from 0, so that the last element in prefix $\alpha$ is in position...
Lemma 5. Let \( p = \alpha \beta^\omega \in SV(\phi)^\omega \) be an ultimately periodic word, and \( \beta = \beta' \beta'' \) for some \( \beta', \beta'' \in SV(\phi)^* \), \( s' \in SV(\phi) \); let \( i \) be the position of \( s' \) in \( \alpha \beta \) (so \( \rho(i) = s' \)). Let \( p_1, p_2 \) any two points of \( P_\rho \) such that \( sv(p_1) = i, sv(p_2) = j \) and \( p_1 \equiv p_2 \). If \( j > i + |\beta| \) and \( p_1 \preceq p_j \) (with \( \preceq \in \{<,\preceq,\approx,\succeq,\succ\} \)), then it is also \( p_1 \preceq p' \), with \( p \equiv p' \) and \( sv(p') = j - |\beta| \).

Proof. First of all, note that, since \( p_1 \equiv p_j \), it is \( \rho(j - |\beta|) = \rho(j) = s' \).

Let us consider the case \( p_i \preceq p_j \). Then, there exist three points \( p_1, p_2, p_3 \) such that:

1. either \( p' \preceq p_1 \), or \( p' \succeq p_1 \)
2. \( p_1 \preceq p_2 \)
3. \( p_2 \preceq p_j \)
4. \( p_1 \preceq p_3 \)
5. \( p_3 \preceq p_1 \)
6. either \( p_3 \preceq p' \), or \( p_3 \succeq p' \).

Figure 4 exemplifies the conditions above. We have two cases. If \( p' \preceq p_1 \), then, from conditions 2 and 3, and the definition of \( \preceq \), we have \( p' \preceq p_j \); since \( p_i \), \( p' \) and \( p_j \) all belong to \( \beta^\omega \) and are such that \( p_i \equiv p' \equiv p_j \), then the same forward path between \( p' \) and \( p_j \) from which it descends \( p' \preceq p_j \) can be iterated starting from \( p_i \), because suffix \( \beta^\omega \) is periodic. Then, \( p_i \preceq p' \). If, instead, \( p' \succeq p_1 \), then, by conditions 5 and 6, and
Proof. Let us assume in \( sv \) labeled with \( q \) such that, for each \( p \) and \( p \), this case it must also be \( p \). If \( p \) \( p \), then if it is \( p \) \( p \), it is also \( p \) \( p \), and the proof is as before. If, instead, it is not \( p \) \( p \), then it must be \( p \). Otherwise, if it is not \( p \) \( p \), then it must be \( p \) \( p \), and in this case it must also be \( p \) \( p \). Then it is also \( p \) \( p \). Otherwise, if it is not \( p \) \( p \), then it must be \( p \) \( p \), and \( p \) is labeled with \( = \), and we have that \( p \) \( p \), not \( p \) \( p \) (hence \( p \) \( p \) by Remark 1), and \( p \) \( p \), which yields a contradiction.

The proofs for cases \( p \) \( p \), \( p \) \( p \), and \( p \) \( p \) are similar. \( \square \)

We immediately have the following corollary, which states that a path looping through \( p \) can be shortened to a single iteration.

**Corollary 3.** Let \( \rho = \alpha \beta^\omega \in SV(\phi)^\omega \), \( p_i \) and \( p_j \) as in Lemma 5. Then it is also \( p_i \approx p_j \), with \( p \equiv p' \) and \( sv(p') = i + |\beta| \).

The following lemma shows that there is an infinite non-strict (resp. strict) forward path in \( \rho \approx (\alpha')s(\beta')s \) if, and only if, there is an infinite non-strict (resp. strict) forward path that loops through symbolic valuation \( s \).

**Lemma 6.** Let \( \rho = \alpha \beta^\omega \in SV(\phi)^\omega \) be an ultimately periodic word, with \( \alpha = \alpha' s \) and \( \beta = \beta' s \). In \( \rho \) there is an infinite non-strict (resp. strict) forward path if, and only if, there is an infinite non-strict (resp. strict) forward path that contains a denumerable set of points \( \{ p_i \}_{i \in N} \) of \( P_\rho \) such that:

1. \( sv(p_0) = |\alpha| - 1 = |\alpha'| \),
2. \( p_i \equiv p_j \) and \( sv(p_i) < sv(p_j) \) for all \( i < j \in N \),
3. \( p_i \approx p_i+1 \) (resp. \( p_i \approx p_i+1 \)) for all \( i \in N \).

**Proof.** Let us assume in \( \rho \) there is an infinite non-strict forward path, and let \( F = \{ f_i \}_{i \in N} \) be the points that it traverses (hence, it is \( f_i \equiv f_{i+1} \) for all \( i \)). Note that \( sv(f_0) \) can be any, not necessarily 0 or \( |\alpha'| \). Since suffix \( \beta^\omega \) is periodic and each arc \( \langle f_i, f_{i+1} \rangle \) in \( F \) connects two points that, for Condition 1 of Definition 7, dist at most \( -|\phi| + |\phi| + 1 \) from one another, then there must be a sequence of points \( Q = \{ q_i \}_{i \in N} \) such that, for each \( q_i \in Q \)

- \( sv(q_{i+1}) > sv(q_i) > |\alpha'| \)
- there is a point \( f_j \in F \) such that \( f_j \) is locally equivalent to \( q_i \)
- \( \rho(sv(q_i)) = s \).
In other words, \( Q \) is made by points of \( F \) (or locally equivalent ones) that belong to one of the instances of symbolic valuation \( s \) in \( \beta^\omega \). For each \( i \in \mathbb{N} \) it is \( q_i \not\preceq q_{i+1} \). Since the number of points in symbolic valuation \( s \) is finite, there must be an element \( q_\bar{i} \in Q \) such that an infinite number of points equivalent to \( q_\bar{i} \) appear in \( Q \). In other words, there is a denumerable sequence \( L = \{l_i\}_{i \in \mathbb{N}} \) such that

- \( l_0 = q_\bar{i} \)
- for all \( i \) it is \( l_i \equiv l_{i+1} \) (also, it is \( sv(l_i) < sv(l_{i+1}) \)). Sequence \( L \) is part of an infinite forward path that starts from \( l_0 \) and visits all \( l_i \). The desired sequence \( \{p_i\}_{i \in \mathbb{N}} \) that satisfies conditions 1-3 is \( L \) translated of \( sv(l_0) - |\alpha'| \), so that it starts from symbolic valuation in position \( |\alpha'| \) (the translation is possible because of the periodicity of \( \beta^\omega \)). Figure 5 shows an example of translation.

The proof in case of strict infinite paths is similar.

A similar lemma holds for backward paths. We have the following result.

**Theorem 4.** Let \( \rho = \alpha\beta^\omega \in SV(\phi)^\omega \) be an ultimately periodic word, with \( \alpha = \alpha' s \) and \( \beta = \beta' s \). Then, there is a non-strict (resp. strict) infinite forward path in \( \rho \) if, and only if, there are two points \( p, p' \) of \( P_{\rho} \) such that \( sv(p) = |\alpha'| \), \( sv(p') = k \), \( p \equiv p' \), and \( p \prec p' \) (resp. \( p \preceq p' \)).

**Proof.** We consider the case for non-strict forward paths, the case for strict ones being similar.

Assume in \( \rho \) there is an infinite non-strict forward path; then, by Lemma 6 there is also an infinite non-strict forward path that contains a denumerable set of points \( \{p_i\}_{i \in \mathbb{N}} \) that satisfies conditions 1-3 of the lemma. Then, from Corollary 3 we immediately have \( p_0 \preceq p' \), with \( p' \equiv p_0 \) and \( sv(p') = |\alpha'| + |\beta'| = k \) (recall that \( |\alpha\beta| = k + 1 \)).

Conversely, assume that there are two points \( p, p' \) such that \( p = (x, |\alpha'|, h) \), \( p' = (x, k, h) \), \( p \equiv p' \), and \( p \prec p' \). By definition of \( p \prec p' \), there exists a finite number of points \( p', p^2, \ldots \) such that \( p \prec p^1 \prec p^2 \ldots \prec p' \). This forward path can be iterated infinitely many times, since \( p \equiv p' \) and the suffix \( \beta \) is repeated infinitely often. Therefore, point \( p \) and points equivalent to \( p \) satisfy conditions 1-3 of Lemma 6. By the same lemma, then, in \( \rho \) there is an infinite non-strict forward path. \( \square \)
Analogously, we can prove the following version of Theorem 4 in case of backward paths.

**Theorem 5.** Let \( \rho = \alpha \beta^\omega \in SV(\phi)^\omega \) be an ultimately periodic word, with \( \alpha = \alpha' s' \) and \( \beta = \beta' s' \). Then, there is a non-strict (resp. strict) infinite backward path in \( \rho \) if and only if, there are two points \( p, p' \) such that \( sv(p) = |\alpha'|, \; sv(p') = k, \; p \equiv p' \), and \( p \succ p' \) (resp. \( p \succ p' \)).

Our condition for the non existence of an arithmetic model for symbolic model \( \rho = \alpha' s(\beta' s)^\omega \) (with \( |\alpha' s\beta' s| = k + 1 \)) if formalized by Formula (1) below; it captures Property 1 and takes advantage of the previous Theorems 4 and 5.

\[
\exists p_1 p_2 p_1' p_2'
\begin{pmatrix}
 p_1 \equiv p_2 \land p_1' \equiv p_2' \land \\
 sv(p_1) = sv(p_1') = |\alpha'| \land sv(p_2) = sv(p_2') = k \land \\
 p_1 \prec p_2 \land p_1' \succ p_2' \land (p_1 \prec p_2 \lor p_1' \succ p_2') \land \\
 (p_1 \prec p_1' \lor p_1' \succ p_1) 
\end{pmatrix}.
\tag{1}
\]

In Formula (1) four conditions are defined, similar to those of Property 1. Informally, Formula (1) says that:

1. there is an infinite forward path \( f \) from \( p_1 \) (this derives from the fact that \( p_1 \prec p_2 \), with \( p_1 \equiv p_2 \), \( sv(p_1) = |\alpha'| \), and \( sv(p_2) = k \));
2. there is an infinite backward path \( b \) from \( p_1' \) (from \( p_1' \succ p_2' \), with \( p_1' \equiv p_2' \), where \( sv(p_1') = |\alpha'| \), and \( sv(p_2') = k \));
3. at least one between \( f \) and \( b \) is strict;
4. between \( p_1 \) and \( p_1' \) there is an edge labeled with \(<\).

In particular, condition 4 of Property 1 is different from condition 4 of Formula (1). In fact, the former one states that for each \( i, j \in \mathbb{N} \), given a forward path \( d \) and a backward path \( e \), whenever \( d(i) \) and \( e(j) \) belong to the same symbolic valuation (i.e., \( |i - j| < |\phi| + |\phi| + 1 \) there is an edge labeled by \(<\) from \( d(i) \) to \( e(j) \)). In other words, this means that point \( p_d \) representing \( d(i) \) and point \( p_e \) representing \( e(j) \) are such that either \( p_d \prec p_e \) or \( p_e \succ p_d \). The next theorem shows that the conditions are nevertheless equivalent when \( \rho = \alpha \beta^\omega \). In fact, whereas Property 1 is defined for a general \( G_\rho \), Formula (1) is tailored to the finite representation of ultimately periodic symbolic models \( \rho = \alpha \beta^\omega \).

**Theorem 6.** Over ultimately periodic symbolic models of the form \( \alpha' s(\beta' s)^\omega \), with \( \alpha, \beta \in SV(\phi)^* \) and \( s \in SV(\phi) \), Property 1 is equivalent to Formula (1).

**Proof.** Let \( \rho = \alpha' s(\beta' s)^\omega \) be an infinite symbolic model and assume that Formula (1) holds in \( \alpha' s\beta' s \). Therefore, by Theorems 4 and 5 there exists a pair of points \( p_1 \) and \( p_1' \), such that \( sv(p_1) = sv(p_1') = |\alpha'| \), visited respectively by an infinite forward path and an infinite backward path, where at least one of the two is strict (because
Since $p_1 \not\sim p_2 \lor p_1' \not\sim p_2'$ holds. Since $p_1 \prec p_1' \lor p_1' \succ p_1$ holds, and $\equiv$ is a congruence for $\prec$, $\succ$, then also $p_2 \prec p_2' \lor p_2' \succ p_2$ hold. Now, consider any two points $u$ and $v$ in $\alpha'\beta'\gamma$, such that $sv(u) = sv(v)$ and $u$ (resp. $v$) belongs to the infinite strict forward (resp. backward) path from $p_1$ (resp. $p_1'$). Then, it is $u \prec p_2$, $v \succ p_2'$, and $p_2 \prec p_2'$ or $p_2' \succ p_2$. Hence, it is also $u \prec v$ or $v \succ u$, i.e., between $u$ and $v$ there is an edge labeled with $\prec$.

Conversely, assume Property [1] holds; then, by Theorems [4] and [5] there are points $p_1, p_1', p_2, p_2'$ such that $sv(p_1) = sv(p_1') = |\alpha'|$, $sv(p_2) = sv(p_2') = k$, $p_1 \equiv p_2$, $p_1' \equiv p_2'$, $p_1 \prec p_2$, $p_1' \succ p_2'$, and $p_1 \prec p_2 \lor p_1' \succ p_2'$ hold. From the proof of Theorem [4] point $p_1$ is equivalent to some point in the original forward path; similarly for point $p_1'$. Then, since $p_1$ and $p_1'$ belong to the same symbolic valuation, by condition 4 of Property [1] they are connected through an edge labeled with $\prec$, i.e., $p_1 \prec p_1'$ or $p_1' \succ p_1$ hold.

The next theorem extends Proposition [5] to constraint system IPC*, which does not benefit from the completion property.

**Proposition 6.** Let $\phi \in CLTLB(D)$ and $D$ be IPC*. Formula $\phi$ is $k$-satisfiable and Formula [7] does not hold if, and only if, there exists a model $\sigma$ such that $\sigma, 0 \models \phi$.

**Proof.** By Theorems [1] and [6] $\phi$ is $k$-satisfiable if, and only if, formula $|\phi|_k$ is satisfiable; in addition, when formula $|\phi|_k$ is satisfiable, it induces a model $\sigma_k$ and a sequence $\alpha\beta$ of symbolic valuations of length $k$ representing an infinite sequence $\rho = \alpha\beta\gamma$ of symbolic valuations such that $\rho \models \phi$. Since Formula [1] does not hold, then by Theorem [6] Property [1] does not hold, hence, by Lemma [2] $\rho$ admits a model $\sigma$ such that $\sigma, 0 \models \phi$.

Conversely, if formula $\phi$ is satisfiable, then automaton $A_\phi$ recognizes models which satisfy condition $C$. Then, a symbolic model $\alpha\beta\gamma \in A(\phi)$ and a model $\sigma_k, 0 \models \alpha\beta\gamma$ can be obtained as in the proof of Proposition [5].

### Bounded Encoding of Formula [1]

The encoding shown afterwards represents, by means of a finite representation, infinite – strict and non strict – paths over infinite symbolic models. As before, we consider models $\rho = \alpha\beta\gamma$ where $\alpha = \alpha'\beta'$ and $\beta = \beta'\gamma$, and we consider the finite sequence of symbolic valuations $\alpha'\beta'\gamma$, of length $k + 1$. We indicate by $P_{\alpha\beta} \subset P_\rho$ the set of points of finite path $\alpha'\beta'\gamma$, of all $p \in P_{\alpha\beta}$, it is $sv(p) \in [0, k]$). We use the points of $P_{\alpha\beta}$ to capture properties of $P_\rho$. To encode the previous formulae into QF-EUD formulae, where $D$ is a suitable constraint system embedding $\mathbb{N}$ and having the successor function plus order $\prec$, we rearrange the formulae above by splitting information, which is now encapsulated in the notion of point, on variables and positions over the model. Predicate $f_{x,y}^\prec : \mathbb{N}^3 \rightarrow \{true, false\}$ for all pairs $x, y \in V \cup const(\phi)$ (resp. $f_{x,y}^{\geq}$) encodes relation $p_1 \prec p_2$ (resp. $p_1 \leq p_2$) where $p_1 = (x, j, h)$ and $p_2 = (y, j, m)$.

\[
\begin{align*}
0 \leq j \leq k & \land h \leq m & f_{x,y}^\prec(j, h, m) & \Leftrightarrow \sigma_k(j + h, x) < \sigma_k(j + m, y) \\
0 \leq j \leq k & \land h > m & f_{x,y}^\prec(j, h, m) & \Leftrightarrow \sigma_k(j + h, x) \leq \sigma_k(j + m, y) \\
0 \leq j \leq k & \land h \leq m & f_{x,y}^{\geq}(j, h, m) & \Leftrightarrow \sigma_k(j + h, x) \geq \sigma_k(j + m, y) \\
0 \leq j \leq k & \land h > m & f_{x,y}^{\geq}(j, h, m) & \Leftrightarrow \sigma_k(j + h, x) > \sigma_k(j + m, y)
\end{align*}
\]
for all \( h, m \in [\{\phi\}, \{\phi\}] \). The value of \( \sigma_k(0+h, x) \) equals the value of term \( \alpha = Y^{|h|}x \), for \( h \in [\{\phi\}, -1] \), or of term \( \alpha = X^h x \), for \( h \in [0, [\phi]] \). For example, \( \sigma_k(0+h, x) \) is \( \alpha(0) \), and \( \sigma_k(k+h, x) \) is \( \alpha(k) \) (see [ArithConstraints]_k in Section 3.3). Constants are implicitly included in the model. For instance, if \( 5 \in const(\phi) \) and \( x \in V \) we have the following formulae \( f_{x,0}^c(j, h, m) \) iff \( \sigma_k(j + h, x) < 5 \) and \( f_{5,x}^c(j, h, m) \) iff \( 5 < \sigma_k(j + m, x) \). When \( x, y \in const(\phi) \) then \( f_{x,y}^c \) iff \( x < y \) and \( f_{x,y}^c \) iff \( x \leq y \) for all \( 0 \leq j \leq k \) and \( h \leq m \); \( \neg f_{x,y}^c \) and \( f_{x,y}^c \) for all \( 0 \leq j \leq k \) and \( h > m \).

Relation \( \preceq \) (resp. \( \preccurlyeq \)) is encoded by uninterpreted predicates \( F_{x,y}^\leq : \mathbb{N}^2 \rightarrow \{true, false\} \) (resp. \( F_{x,y}^\leq : \mathbb{N}^2 \rightarrow \{true, false\} \)) for all pairs of variables \( x, y \in V \cup const(\phi) \). To build in practice \( \preceq \) (resp. \( \preccurlyeq \)) through \( F^\prec \) (resp. \( F^\preceq \)), over points of the symbolic model \( \alpha's beta's \), we construct the transitive closure of \( F^\prec \) (resp. \( F^\preceq \)) explicitly. Starting from \( \rho(0) \), we propagate the information about relations \( \prec \) and \( \preceq \) that are represented by \( f^\prec \) and \( f^\preceq \) among all points representing variables of model \( \rho \). In fact, it is immediate to show that \( p_1 \preceq p_2 \) holds if, and only if, there is a point \( p \) such that either \( p_1 \prec p \) and \( p \preceq p_2 \) or \( p_1 \preceq p \) and \( p \preceq p_2 \) (note that \( p \) cannot be locally equivalent to both \( p_1 \) and \( p_2 \), but it can be locally equivalent to one of them). Similarly for the other relations. Figure 6 provides a graphical representation for \( \preceq \). Formulae defining \( F_{x,y}^\leq \) and \( F_{x,y}^\geq \) are the following:

\[
F_{x,y}^\leq(j, h, i, m) \Leftrightarrow \bigvee_{z \in V} \bigwedge_{u = [\phi]} \left[ [\phi] \bigvee_{z \in V} f_{x,z}^\leq(j, h, u) \wedge F_{x,y}^\leq(j, u, i, m) \bigvee \bigwedge_{f_{5,z}^\leq(j, h, u)} \wedge F_{x,y}^\leq(j, u, i, m) \right]
\]

(2)

\[
F_{x,y}^\geq(j, h, i, m) \Leftrightarrow \bigvee_{z \in V} \bigwedge_{u = [\phi]} \left[ [\phi] \bigvee_{z \in V} f_{x,z}^\geq(j, h, u) \wedge F_{x,y}^\geq(j, u, i, m) \bigvee \bigwedge_{f_{5,z}^\geq(j, h, u)} \wedge F_{x,y}^\geq(j, u, i, m) \right]
\]

(3)

for all \( j, i \in [0, k] \) with \( j < i \) and for all \( h, m \in [\{\phi\}, \{\phi\}] \) such that \( i + m - (j + h) > -[\phi] + [\phi], h = [\phi], (x = z) \Rightarrow (h \neq u) \) and for all pairs \( x, y \in V \cup const(\phi) \).
When \( j = i \in [0, k] \) and \( h \leq m \), with \( h, m \in [[\phi], [\phi]] \):
\[
\begin{align*}
F_{x,y}^{<}(j, h, i, m) & \iff F_{x,y}^{<}(j, h, j, m) \\
F_{x,y}^{\preceq}(j, h, i, m) & \iff F_{x,y}^{\preceq}(j, h, j, m)
\end{align*}
\]
When \( j + h > i + m \):
\[
\begin{align*}
\neg F_{x,y}^{<}(j, h, i, m) \\
\neg F_{x,y}^{\preceq}(j, h, i, m)
\end{align*}
\]
Figure 7 shows how predicate \( F_{x,x}^{<}(i, 0, j, 1) \) is defined as conjunction of local relation \( f_{x,y}^{<}(i, 0, 1) \) and of \( F_{y,x}^{<}(i, 1, j, 1) \).

The following formula \(|\text{CongruenceConstraints}|_k\) defines congruence classes of locally equivalent points for relations \(~\preceq\!, \preceq!\) captured by predicates \( F^{<} \) and \( F^{\preceq} \). In fact, observe that, since from \( p_1 \preceq p_2 \) we obtain \( p'_1 \preceq p'_2 \), for all \( p'_1 \) (resp. \( p'_2 \)) that is locally equivalent to \( p_1 \) (resp. \( p_2 \)), then, in general, the congruence extends to \( \preceq! \): i.e., from \( p_1 \preceq! p_2 \) we obtain \( p'_1 \preceq! p'_2 \) for all \( p'_1, p'_2 \) locally equivalent to \( p_1, p_2 \). An analogous argument holds for \(~\succ\!, \succ!\) and \(~\prec\!, \prec!\).

\[
\begin{array}{ccc}
\begin{array}{llll}
i \in [1, k] & | & m \in [[\phi], [\phi]] & | & j \\
F_{x,y}^{<}(j, h, i, m) & \iff & F_{x,y}^{<}(j, h, j, m) & h \in [[\phi] + 1, [\phi]] & [0, i - 1] \\
F_{x,y}^{\preceq}(j, h, i, m) & \iff & F_{x,y}^{\preceq}(j, h, j, m) & h \in [[\phi], [\phi] - 1] & [1, i]
\end{array} \\
\begin{array}{llll}
j \in [0, k - 1] & | & h \in [[\phi], [\phi]] & | & i \\
F_{x,y}^{<}(j, h, i, m) & \iff & F_{x,y}^{<}(j, h, i + 1, m - 1) & m \in [[\phi] + 1, [\phi]] & i \in [j, k - 1] \\
F_{x,y}^{\preceq}(j, h, i, m) & \iff & F_{x,y}^{\preceq}(j, h, i - 1, m + 1) & m \in [[\phi] + 1, [\phi] - 1] & i \in [j + 1, k]
\end{array}
\end{array}
\]

Predicates \( b_{x,y}^{>}, b_{x,y}^{\geq} \) for local backward paths \( \succ, \succeq \), predicates \( B_{x,y}^{>}, B_{x,y}^{\geq} \) for backward paths \( \preceq, \preceq! \) and congruence among points are defined similarly. For brevity, we only show the definition of \( b_{x,y}^{>}, b_{x,y}^{\geq} \) and \( b_{x,y}^{\leq} \), the others are straightforward.

\[
\begin{array}{ccc}
\begin{array}{lll}
0 \leq j \leq k & \text{and} & h \leq m \\
b_{x,y}^{>}(j, h, m) & \iff & \sigma_k(j + h, x) > \sigma_k(j + m, y) \\
b_{x,y}^{\geq}(j, h, m) & \iff & \sigma_k(j + h, x) \geq \sigma_k(j + m, y)
\end{array}
\end{array}
\]

\[
\begin{array}{ccc}
\begin{array}{lll}
0 \leq j \leq k & \text{and} & h > m \\
\neg b_{x,y}^{>}(j, h, m) & \iff & \sigma_k(j + h, x) \leq \sigma_k(j + m, y) \\
\neg b_{x,y}^{\geq}(j, h, m) & \iff & \sigma_k(j + h, x) < \sigma_k(j + m, y)
\end{array}
\end{array}
\]

32
for all $h, m \in [[\phi], [\phi]]$. When both $x, y \in const(\phi)$ then $b_{x,y}^j(j, h, m)$ iff $x > y$ and $b_{x,y}^{-j}(j, h, m)$ iff $x \geq y$ for all $0 \leq j \leq k$ and $h \leq m$; $-b_{x,y}^j(j, h, m)$ and $-b_{x,y}^{-j}(j, h, m)$ for all $0 \leq j \leq k$ and $h > m$.

Finally, the condition of existence defined by Formula (1) is encoded by the following QF-EU formula. The condition is parametric with respect to a pair of variables $x, x' \in V \cup const(\phi)$. The condition is meaningful only if $x \neq x'$ and if either $x \notin const(\phi)$ or $x' \notin const(\phi)$. In fact, a constant value never generates a strict (forward or backward) path; therefore, two constants can not satisfy the condition of non-existence of an arithmetical model. Formula $C_{x,x'}$ below captures the existence in $\rho([\alpha'])$ of a strict relation $<$ between two points, one of a forward and one of backward path, which involve variables $x$ and $x'$. Variable $loop$ has already been introduced in Section 5 and defines the position where, in $\alpha \beta$, suffix $\beta$ starts (as already explained $|\alpha'| = loop - 1$).

$$C_{x,x'} := \bigvee_{h,h' \in [[\phi],[\phi]]} \left( \left( F_{x,x}^\leq (loop - 1, h,k,h) \land B_{x,x}^< (loop - 1, h',k,h') \right) \right) \vee \left( F_{x,x}^> (loop - 1, h,k,h) \land B_{x,x}^\geq (loop - 1, h',k,h') \right) \land \left( f_{x,x}^\leq (loop - 1, h,h') \lor b_{x',x'}^> (loop - 1, h',h) \right)$$

In formula $C_{x,x'}$, we use explicitly points that were symbolically represented in Formula (1): $p_1 = (x, loop - 1, h)$, $p'_1 = (x', loop - 1, h')$, $p_2 = (x,k,h)$, $p'_2 = (x',k,h')$. It is immediate to see that formula $f_{x,x'}^\leq (loop - 1, h,h') \lor b_{x',x'}^> (loop - 1, h',h)$ encodes $p_1 < p'_1 \lor p'_1 > p_1$ of Formula (1) and formula $F_{x,x}^\leq (loop - 1, h,k,h) \land B_{x,x}^> (loop - 1, h',k,h')$ encodes $p_1 \leq p_2 \land p'_1 \leq p'_2 \land p_1 \leq p_2$ (similarly for formula $F_{x,x}^< (loop - 1, h,k,h) \land B_{x,x}^\geq (loop - 1, h',k,h')$).

The existence condition of an arithmetical model is captured by the formula:

$$\bigwedge_{x,x' \in V \cup const(\phi)} \neg C_{x,x'} \land x \neq x', x \notin const(\phi) \lor x' \notin const(\phi)$$

Given a CLTLB(IPC$^*$) formula $\phi$, the satisfiability of $\phi$ is reduced to the satisfiability of the following QF-EU(D) formula:

$$|\phi|_k \land (4)$$

If Formula (5) is unsatisfiable, then either $\phi$ does not admit symbolic models, or none of its symbolic models admits arithmetical models. Conversely, if Formula (5) is satisfiable, then there is a symbolic model $\rho$ of $\phi$ for which condition (4) holds, hence $\rho$ admits an arithmetical model and $\phi$ is satisfiable.

### 4.1 Simplifying the condition of existence of arithmetical models

In this section, we relax the condition of existence of an arithmetical model $\sigma$ for sequences of symbolic valuations of CLTLB(IPC$^*$) formulae. In fact, Property (1) is
stronger than necessary in those cases in which not all variables appearing in a formula \( \phi \) are compared against each other. Consider for example the following formula

\[
G(x < Xx \land \neg(y < Xy))
\]

which enforces strict increasing monotonicity for variable \( x \) and decreasing monotonicity for variable \( y \). Figure 8 shows a symbolic model for Formula (6) which does not admit arithmetic model, as it does not satisfy Property 1 (in fact, the strict forward path that visits all points \( \{(x, i, 0)\}_{i \in \mathbb{N}} \) and the strict backward path that visits all points \( \{(y, i, 0)\}_{i \in \mathbb{N}} \) are such that, for all \( i \), \( (x, i, 0) \prec (y, i, 0) \)). However, in Formula (6) \( x \)

and \( y \) are not compared, neither directly, nor indirectly, so if we disregard the relations between them in the symbolic model of Figure 8 and produce an assignment of the variables that only respects the relations between variables that are actually compared in the formula (i.e., \( x \) with itself, and \( y \) with itself) we obtain an arithmetic model for Formula (6). Figure 9 shows a “weaker” version of the symbolic model of Figure 8 one that is more concise to encode into QF-EU(D) formula than the maximally consistent one, as it does not contain any comparison between unrelated terms.

\[
\begin{array}{c}
x < \ldots < < < < \ldots \\
\wedge \wedge \wedge \wedge \\
0 = 1 > 2 = 3 > 4 \ldots \\
y
\end{array}
\]

Figure 9: A weak symbolic model for Formula (6).

To characterize sequences of symbolic valuations which do not take into account relations among variables that are not compared with each other in a formula \( \phi \), we first remark that \( \phi \) induces a finite partition \( \{V_1, \ldots, V_h\} \) of set \( V \) such that \( x, y \in V_i \) if and only if there is an IPC constraint \( R(X_i^x, X_j^y) \) occurring in \( \phi \), for some \( i, j \in \mathbb{Z} \) (where we write \( X_i^x \), with \( n \geq 0 \), instead of \( Y^n \)). Then, we introduce the notions of weak symbolic valuation and of sequence of weak symbolic valuations.

**Definition 11.** Given a symbolic valuation \( sv \in SV(\phi) \), its weak version \( \overline{sv} \) is obtained by removing from \( sv \) all relations \( R(X_i^x, X_j^y) \) where \( x \in V_l \) and \( y \in V_t \) with \( l \neq t \). We similarly define the weak version \( \overline{\rho} \) of a sequence \( \rho \) of symbolic valuations.
Given a CLTLB(IPC*) formula $\phi$, we indicate with $SV_w(\phi)$ the set of all its weak symbolic valuations. A weak symbolic model $\overline{\sigma} \in SV_w(\phi)^\omega$ of $\phi$ is a sequence of weak symbolic valuations such that $\overline{\sigma}, 0 \models \phi$. Given $\rho \in SV(\phi)^\omega$ and its weak version $\overline{\rho}$, $G_{\overline{\rho}}$ is the subgraph of $G_\rho$ obtained by removing all arcs between points $p = (x, j, h)$, $p' = (y, i, m)$ such that $x \in V_i$, $y \in V_i$, and $l \neq t$.

The next lemma shows that focusing on weak symbolic valuations is enough to determine whether symbolic models for $\phi$ exist or not.

**Lemma 7.** Let $\phi$ be a CLTLB(IPC*) formula. Given $\rho \in SV(\phi)^\omega$ such that $\rho, 0 \models \phi$, it is also $\overline{\rho}, 0 \models \phi$. Conversely, given a sequence $\nu \in SV_w(\phi)$ of weak symbolic valuations, if $\nu, 0 \models \phi$, then for any $\rho \in SV(\phi)$ such that $\overline{\nu} \models \nu$ it is also $\rho, 0 \models \phi$.

**Proof.** Assume that $\rho \models \phi$. We only need to focus on the base case, as the inductive case is trivial. For all $i$ and $R(\alpha_1, \alpha_2)$ occurring in $\phi$, $\rho, 0 \models R(\alpha_1, \alpha_2)$ if, and only if, $R(\alpha_1, \alpha_2) \in \rho(i)$.

Since $R(\alpha_1, \alpha_2)$ occurs in $\phi$ then, by Definition 11, it is also $R(\alpha_1, \alpha_2) \in \overline{\rho}(i)$, hence $\overline{\rho}, 0 \models \phi$.

The converse case is similar. If $\nu \in SV_w(\phi)$ is such that $\nu, 0 \models \phi$, then for all $i$ and $R(\alpha_1, \alpha_2)$ that occurs in $\phi$ it is $\nu, 0 \models R(\alpha_1, \alpha_2)$ if, and only if, $R(\alpha_1, \alpha_2) \in \nu(i)$; in addition, for any $\rho$ such that $\overline{\rho} = \nu$ we have $R(\alpha_1, \alpha_2) \in \rho(i)$ if, and only if, $R(\alpha_1, \alpha_2) \in \nu(i)$.

Finally, $\nu, 0 \models \phi$ implies $\rho, 0 \models \phi$.

We have the following variant of Lemma 2, which defines a condition of existence of arithmetical models for symbolic ones that is checked on their weak counterparts.

**Lemma 8.** Let $\phi$ be a CLTLB(IPC*) formula. Given an ultimately periodic, locally consistent sequence $\rho \in SV(\phi)^\omega$ of symbolic valuations, if there is $\sigma : \mathbb{Z} \times V \to D$ such that $\sigma, 0 \models \rho$, then Property 1 holds for graph $G_{\overline{\rho}}$. Conversely, if $\nu \in SV_w(\phi)^\omega$ is an ultimately periodic, locally consistent sequence of weak symbolic valuations such that Property 1 holds for graph $G_{\overline{\rho}}$, then there are $\sigma, \rho$ such that $\overline{\nu} \models \nu$ and $\sigma, 0 \models \phi$.

**Proof.** If there is $\sigma$ such that $\sigma, 0 \models \rho$ then, by Lemma 2 Property 1 holds for $G_{\overline{\rho}}$. Since $G_{\overline{\rho}}$ is a subgraph of $G_\rho$, a fortiori Property 1 holds for $G_{\overline{\rho}}$.

Conversely, if Property 1 holds for $G_{\overline{\rho}}$, then each set of variables $V_i$, with $i \in \{1...n\}$, in which $V$ is partitioned induces an ultimately periodic sequence $\nu_{V_i}$ of symbolic valuations that only include constraints on $V_i$, such that its graph $G_{\nu_{V_i}}$ is not connected to any other graph $G_{\nu_{V_j}}$, for $j \neq i$. Then, Lemma 2 can be applied to $\nu_{V_i}$, which then admits an arithmetic model $\sigma_{V_i} : \mathbb{Z} \times V_i \to D$. By definition, each $\sigma_{V_i}$ assigns a different set of variables, so the complete arithmetic model $\sigma$ is simply the union of all $\sigma_{V_i}$. By Lemma 3 $\sigma$ induces a sequence of symbolic valuations $\rho$, and $\sigma, 0 \models \rho$, $\overline{\rho} \models \nu$ by construction.

Thanks to Lemmata 7 and 8 in Formula 1 and in the corresponding QF-EU(D) encoding of Formula 4 we can focus only on relations between points that belong to the same set $V_i$. 

35
5 Complexity and Completeness

Complexity

In the following we provide an estimation of the size of the formulae constituting the encoding of Section 3.3 including, where they are needed, the constraints of Section 4.

The encoding of Section 3.3 is linear in the size of the formula \( \phi \) (and of the bound \( k \)). In fact, if \( m \) is the total number of subformulae and \( n \) is the total number of temporal operators \( U \) and \( R \) occurring in \( \phi \), the QF-EU encoding requires \( n + 1 \) integer variables (one each for loop and the \( \psi \)'s) and \( m \) unary predicates (one for each subformula in \( cl(\phi) \)).

The total size of the formulae in Section 4 is polynomial in bound \( k \), in the cardinality of the set of variables and constants, and in the size of symbolic valuations. In fact, the encoding of the condition for the existence of an arithmetical model requires a QF-EU(\( \mathbb{N}, <, = \)) formula of size quadratic in the length \( k \), cubic in the number \( |V| \) of variables, and double quadratic in the size of symbolic valuations.

Let \( \lambda \) be the size \( \lambda = \lceil \phi \rceil - \lfloor \phi \rfloor + 1 \) of symbolic valuations and \( V' \) be the set \( V \cup const(\phi) \). The total number of non-trivial predicates \( f_{x,y}^\leq, f_{x,y}^< \) (resp. \( b_{x,y}^\geq, b_{x,y}^> \)), i.e., those where \( h \leq m \), is defined by the following parametric formula (where \( a, b \) are the sets to which \( x,y \) belong, respectively):

\[
N(a, b) = (k + 1) \sum_{i=1}^{\lambda} |a| \cdot ((\lambda - i) + (|b| - 1) \cdot (\lambda - i + 1))
\]

\[
= (k + 1) \left( |a||b| \frac{\lambda(\lambda + 1)}{2} - |a|\lambda \right).
\]

Each predicate has fixed dimension and the number of non-trivial ones results from the sum of the following three cases:

- \( x, y \in V \), which is \( N(V, V) \)
- \( x \in V, y \in const(\phi) \), which is \( N(V, const(\phi)) \)
- \( x \in const(\phi), y \in V \), which is \( N(const(\phi), V) \).

that is bounded by \( N_{local} = N(V', V') \leq (k + 1)|V'|^2\lambda^2 \).

To compute the size of formulae defining \( F_{x,y}^\leq, F_{x,y}^< \) (resp. \( B_{x,y}^\geq, B_{x,y}^> \)) we first determine the number of pairs of points for which \( F_{x,y}^\leq(j, h, i, m) \) is not trivially false. The following function \( N_{p,p'} \)

\[
N_{p,p'} = |V'| \sum_{i=[\phi]}^{k+[\phi]} |V'|(k + [\phi] - i) = |V'|^2 \sum_{i=0}^{k+\lambda-1} i = |V'|^2 \frac{(k + \lambda - 1)(k + \lambda)}{2}
\]

corresponds to the number of pairs of points \( p, p' \) that generate non-trivial predicates \( F_{x,y}^\leq, F_{x,y}^< \) (resp. \( B_{x,y}^\geq, B_{x,y}^> \)) because their position is such that \( sv(p_1) + shift(p_1) \leq
defining predicates requires at most |in the worst case (i.e., for points that do not belong to the same symbolic valuation), F the number of subformulae involved in their definition. We consider only the case for λ longs to size of (non-trivial) formulae (2)-(3) defining F sv possible pair h,h because the others have the same (worst) complexity. Each Formula (2) involves, in the worst case (i.e., for points that do not belong to the same symbolic valuation), |V| − 1 variables z ∈ V with respect to λ different positions u. Then, an instance of (2) requires at most (|V| − 1)λ disjuncts. The upper bound for the total size of all formulae defining predicates F ≤, F < (resp. B ≥, B ≧) is

\[ N_{far} = 2(|V| − 1)λ ≤ λ|V||V'|^2(k + λ)^2 \leq λ|V'|^3(k + λ)^2. \]

The analysis of formulae |CongruenceConstraints|k shows that each point belongs to λ symbolic valuations (e.g., if [φ] = 0, [φ] = −1, then λ = 2, and points (x, 4, 1) and (x, 5, 0) correspond to the same element), and for all pairs p1, p2 we define the consistency of the definition of predicate F ≤, F < among the λ points corresponding to p1 and the λ points corresponding to p2. Therefore, we need at most

\[ N_{CC} = 4|V'|^2 \sum_{i=1}^{k} λ^2 i ≤ 4λ^2|V'|^2k^2 \]

constraints |CongruenceConstraints|k, where each constraint has fixed dimension.

Finally, predicate Cx,x′ appears in Formula (4) once for each of the |V'||V|λ^2 pairs of points x, x′. In addition, each instance of Cx,x′ has λ^2 disjuncts, one for each possible pair h, h′ ∈ [[φ], [φ]]. Therefore, the total size of Formula (4) is \[ N_{C} = |V'||V|\lambda^4. \]

Finally, the complete set of formulae that we require to capture the existence condition of arithmetical models over discrete domains has the following total size:

\[ 4N_{local} + 4N_{far} + 4N_{CC} + N_{C} \leq 4(k + 1)|V'|^2λ^2 + 4λ|V'|^3(k + λ)^2 + 16λ^2|V'|^2k^2 + |V||V'|\lambda^4. \]

In conclusion, for a given formula φ, the parameters λ and |V'| are fixed, hence the size is \( O(k^2) \).

**Completeness**

Completeness has been studied in depth for Bounded Model Checking. Given a state-transition system M, a temporal logic property φ and a bound \( k > 0 \), BMC looks for a witness of length \( k \) for \( \neg φ \). If no witness exists then length \( k \) may be increased and BMC may be reapplied. In principle, the process terminates when a witness is found or when \( k \) reaches a value, the completeness threshold (see Definition 4), which guarantees that if no counterexample has been found so far, then no counterexample disproving property \( φ \) exists in the model. For LTL it is shown that a completeness threshold always exists; [Clarke et al., 2004] shows a procedure to estimate an over-approximation of the value, by satisfying a formula representing the existence of an accepting run of the product automaton \( M \times B_{\neg φ} \), where \( B_{\neg φ} \) is the Büchi automaton for \( \neg φ \) and \( M \) is the system to be verified.
In [Bersani et al.] (2011) we have already given a positive answer to the problem of whether there exists a completeness threshold for the satisfiability problem for CLTLB(D), provided that ultimately periodic symbolic models of the form $\alpha_0\beta^\omega$ of CLTLB(D) formulae admit an arithmetic model. By the results of Section 2.4.1 this occurs when the constraint system D has the completion property, or when condition C holds. In [Bersani et al.] (2011) we used a mixed automata- and logic-based approach to show how completeness can be achieved for the satisfiability problem. In that approach automata $A_C$ and $A_\ell$ described in Section 2.4 are represented through CLTLB(D) formula $\phi_{A_C}$ and $\phi_{A_\ell}$, respectively, described below. More precisely, formula $\phi_{A_C}$ captures the runs of automaton $A_C$, and similarly for $\phi_{A_\ell}$ and $A_\ell$. Then, checking the satisfiability for $\phi$ is reduced to studying a finite amount of $k$-satisfiability problems of formula $\phi \land \phi_{A_C} \land \phi_{A_\ell}$ for increasing values of $k$. Automaton $A_\ell$ recognizes sequences of locally consistent symbolic valuations, so its runs are the models of formula $\phi_{A_\ell} := G(\bigvee_{1}^{m} SV)$.

Since the bounded representation of formula (see Section 3.3) is not contradictory (i.e., two consecutive symbolic valuations are satisfiable when they are locally consistent), the previous formula exactly represents words of $L(A_\ell)$. Formula $\phi_{A_C}$, instead, is derived from automaton $A_C$, by means of the translation in [Sistla and Clarke (1985)]. Automaton $A_C$ is built by complementing automaton $A_{\omega_{C}}$ Safra (1988), recognizing the complement language of $L(A_{C})$, which is obtained according to the procedure proposed in [Demri and D’Souza (2007)]. Finally, to check the satisfiability of $\phi$ we verify whether formula $\phi \land \phi_{A_C} \land \phi_{A_\ell}$ is $k$-satisfiable, with $k \in \mathbb{N}$. The existence of a finite completeness threshold for the procedure above is a consequence of the existence of automaton $A_{\omega_{C}}$ (see Section 2.4) recognizing symbolic models of $\phi$, and of Lemma 3 and Proposition 2. Let $rd(A_{\phi})$ be the recurrence diameter of $A_{\phi}$, i.e., the longest loop-free path in the automaton that starts from an initial state [Kroening and Strichman (2003)]. Then, if formula $\phi \land \phi_{A_C} \land \phi_{A_\ell}$ is not $k$-satisfiable for all $k \in [1, rd(A_{\phi}) + 1]$, then there is no ultimately periodic symbolic model $\rho$ such that both $\rho, 0 \models \phi$ and there exists an arithmetic model $\sigma$ with $\sigma, 0 \models \rho$. Hence, formula $\phi$ is unsatisfiable. Otherwise, we have found an ultimately periodic symbolic model $\rho$ of length $k > 0$ which admits an arithmetic model $\sigma$. From the $k$-bounded solution, we have a symbolic model $\rho = \alpha^\omega$ and its bounded arithmetic model $\sigma_k$. The infinite model $\sigma$ is built from $\sigma_k$ by iterating infinitely many times the sequence of symbolic valuations in $\beta$. Therefore, the completeness bound for BSP of CLTLB(D) formulae is defined by the recurrence diameter of $A_{\phi}$.

Thanks to the results of the previous sections, we can simplify the method presented in [Bersani et al.] (2011). We avoid the construction of automaton $A_{\omega_{C}}$ through Safra’s method and the construction of set $SV(\phi)$. In particular, we take advantage of the definition of $k$-bounded models of $\phi$. By Lemma 4 a finite sequence $\sigma_k$ of $D$-valuations induces a unique locally consistent sequence of symbolic valuations $\rho$, such that $\sigma_k, i \models \rho(i)$, for all $i \in [0, k]$. Therefore, we do not need to precompute set $SV(\phi)$ of symbolic valuations and formula $\phi_{A_\ell}$ is no longer needed to obtain a finite locally consistent sequence of symbolic valuations. If $\phi$ is a formula of CLTLB(D) and $D$ has the completion property, we can simply solve $k$-satisfiability problems for $\phi$ instead of $\phi \land \phi_{A_\ell}$; when $D$ does not have the completion property, Formula 1 allows us to avoid the construction of $A_{\omega_{C}}$. In the first case, by Theorems 1–3 and Proposition 5 $\phi|_k$ is satisfiable if, and only if, there is an ultimately periodic run $\alpha^\omega$ which is rec-
ognized by automaton $A_s \times A$. In the second case, Proposition 6 guarantees that $|\phi|_k$ is satisfiable and Formula (1) does not hold if, and only if, $\phi$ is satisfiable. Therefore, model $\alpha \beta \omega$ obtained by solving the $k$-satisfiability problem belongs to the language recognized by automaton $A_s \times A_s$ and also to the one recognized by $A_C$.

The completeness property still holds without the explicit representation of automata $A_s$ and $A_C$ in the formula we check for satisfiability. Since the role of Formula (1) is to filter, by eliminating edges in the automaton, some of the symbolic models of $\phi$ which, in turn, by Theorems 1–3 correspond to the runs of automaton $A_s \times A_s$, the completeness threshold for our decision procedure can be over-approximated by the recurrence diameter of $A_s \times A_s$, which is at most exponential in the size of $\phi$. Since the number of control states of automaton $A_s$ is at most $O(2^{|\phi|})$, a rough estimation for the completeness threshold is given by the value $|SV(\phi)| \cdot 2^{|\phi|}$. The number of symbolic valuations $|SV(\phi)|$ is, in the worst case, exponential in the size of formula $\phi$.

6 Applications of k-bounded satisfiability

The decision procedure described in this paper has been implemented in our bounded satisfiability checker Zot (http://zot.googlecode.com). The aeZot plug-in of Zot solves $k$-satisfiability for CLTLB over Quantifier-Free Presburger arithmetic (QFP), of which IPC$^*$ is a fragment, but it also supports the constraint system $(\mathbb{R},<,=)$. Even if constraint systems like IPC$^*$, or fragments thereof, do not provide a counting mechanism (provided, for instance, through the addition of functions like $+$ in QFP), they can still be used to represent an abstraction of a richer transition system. In fact, functions like addition, or in general relations over counters which embed a counting mechanism, make the satisfiability problem of CLTLB undecidable (see (Demri and D’Souza, 2007, Section 9.3)).

We next exemplify the use of the CLTLB logic to specify and verify systems behavior, thus highlighting the applicability of the approach.

We use CLTLB over $(D,<,=)$ to specify a sorting process of a sequence of fixed length $N$ of values in $D$. Let $v \in D^N$ be the (initial) vector that we want to sort and $a \in D^N$ be the vector during each step of sorting. We write $v(i)$ for the $i$-th component of $v$, $1 \leq i \leq N$. Notice that we will use the notation $a(i)$, which, strictly speaking, is not a CLTLB term; however, since the length of the array is fixed, we can use $N$ variables $a_i$ to represent the elements of $a$, one for each $a(i)$. Then, in the following, if $a(i)$ is replaced with $a_i$, one obtains CLTLB$(D, <, =)$ formulae. We define a set of formulae representing a sorting process which swaps unsorted pairs of values at some nondeterministically chosen position in the vector (we report here only the most relevant formulae). A variable $p \in [0, N - 1]$ stores the position of elements which are a candidate pair for swapping; i.e., $p = i$ means that $a(i)$ is swapped with $a(i + 1)$, while $p = 0$ means that no elements are swapped ($0$ is not a position of the vector). A nondeterministic algorithm can swap arbitrarily two elements in $[1, N]$; then, the only constraint on variable $p$ is that it is $0 \leq p < N$, i.e.: $G(p < N \land p \geq 0)$. An unsorted
A swap between two adjacent positions of \( a \) is formalized by the following formula:

\[
G \left( \bigwedge_{i \in [1,N-1]} p = i \Rightarrow Xa(i) = a(i+1) \land Xa(i+1) = a(i) \right).
\]

Vector \( a \) is unchanged when no pairs are candidate for swapping: \( G(p = 0 \Rightarrow \bigwedge_{i \in [1,N]} (a(i) = Xa(i))) \). Various properties of the algorithm have been verified through the \( ae^2Zot \) plugin of the Zot tool, e.g., whether there exists a way to sort array \( a \) within \( k \) steps (with \( k \) the verification bound), which is formalized by the following formula:

\[
F \left( \bigwedge_{i \in [1,N-1]} (a(i) \leq a(i+1)) \land \bigwedge_{i \in [1,N]} \bigvee_{j \in [1,N]} (a(i) = v(j)) \right).
\]

## 7 Related works

For some constraint system \( D \) more expressive than IPC*, the future fragment CLTL(D) can encode runs of Minsky machines, a class of Turing-equivalent two-counter automata. Minsky machines are finite state automata endowed with two nonnegative integer counters \( c_1, c_2 \) which can be either incremented or decremented by 1 and tested against 0 over transitions. For example, to represent increment and decrement instructions the grammar of formulae \( \xi \) of IPC* can be enriched with formulae of the form \( x < y + d \), where \( d \in D \) and \( x, y \) are variables (these correspond to difference logic – DL – constraints). Hereafter, we write CLTL\( _A(\mathcal{D}) \) to denote the language of CLTL formulae such that the cardinality of \( V \) is \( a \) and \( \lceil \phi \rceil \) is \( b \) (while \( \lfloor \phi \rfloor \) is of course 0).

The first undecidability result for the satisfiability of CLTL is given by Comon and Cortier (Comon and Cortier, 2000, Theorem 3) who show that halting runs of a Minsky machine can be encoded into CLTL\( _A(\mathcal{D}) \) formulae where one auxiliary counter encodes control states of the system labeling instructions. Therefore, the satisfiability problem for CLTL\( _A(\mathcal{D}) \) is \( \Sigma_1^1 \)-hard. The authors suggest a way to regain decidability by means of a syntactic restriction on formulae including the U temporal operator. The “flat” fragment of CLTL\( _A(\mathcal{D}) \) consists of CLTL formulae such that subformula \( \phi \) of \( \phi U \psi \) is \( T, \bot \) or a conjunction \( \zeta_1 \land \cdots \land \zeta_m \) where \( \zeta_i \in \mathcal{D} \). The fragment has a nice correspondence with a special class of counter system (flat relational counter system) with Büchi acceptance condition, for which the emptiness problem is decidable. Satisfiability is undecidable also in the case of CLTL\( _2(\mathcal{D}) \) and CLTL\( _2(\mathcal{D}) \). In fact, even though CLTL\( _2(\mathcal{D}) \) has only one variable, it is expressive enough to encode runs of Minsky machines: models of CLTL\( _2(\mathcal{D}) \) formulae can represent counter \( c_1 \) at even
positions and counter $c_2$ at odd positions. The recurrence problem for nondeterministic Minsky machines, which is $\Sigma_1^1$-hard [Alur and Henzinger (1994)], can be reduced to the satisfiability problem for $\text{CLTL}_2^\omega(\text{DL})$, which then results $\Sigma_1^1$-hard. From the previous undecidability results, the satisfiability problem for the CLTL language over two integer variables $\text{CLTL}_2^\omega(\text{DL})$ is $\Sigma_1^1$-hard. In fact, formulae of $\text{CLTL}_2^\omega(\text{DL})$ can be syntactically translated to formulae of $\text{CLTL}_2^\omega(\text{DL})$ by means of a map $f$ such that $\phi$ belonging to $\text{CLTL}_2^\omega(\text{DL})$ is satisfiable if, and only if, $f(\phi)$ belonging to $\text{CLTL}_2^\omega(\text{DL})$ is satisfiable. Both the languages $\text{CLTL}_1^\omega(\text{DL})$ and $\text{CLTL}_2^\omega(\text{DL})$ are also $\Sigma_1^1$-complete by reducing the $\Sigma_1^1$-hard model-checking problem to satisfiability.

The satisfiability (and model-checking) problem for CLTL over structure $(D, <, =)$ with $D \in \{\mathbb{N}, \mathbb{Z}, \mathbb{Q}, \mathbb{R}\}$ is studied in Demri and D’Souza (2007), and for IPC* in Demri and Gascon (2007). Decidability of the satisfiability problem for the above cases is shown by means of an automata-based approach similar to the standard case for LTL.

Satisfiability for $\text{CLTL}^\omega_2^\omega(\text{IPC})$ and $\text{CLTL}^\omega_2^\omega(\text{IPC})$ over $\mathbb{N}, \mathbb{Z}, \mathbb{Q}, \mathbb{R}$ is obtained by Demri and Gascon in Demri and Gascon (2005) by reducing it to the emptiness of Büchi automata. Given a CLTL formula $\phi$, it is possible to define an automaton $A_\phi$ such that $\phi$ is satisfiable if, and only if, $\mathcal{L}(A_\phi)$ is not empty. Since the emptiness of $\mathcal{L}(A_\phi)$ in the considered structures is decidable with PSPACE upper bound (in the dimension of $\phi$), then the satisfiability problem is also decidable with the same complexity. We remark that the notion of symbolic valuation in that work is different from the one we adopted in Definition 1. Since the procedure is purely symbolic, constraints representing equality relation $x = d$ and constraints of the form $x \equiv c$, with $d, c \in D$, are explicitly considered, as no arithmetical model $\sigma$ is available. A symbolic valuation is defined there as a triple $\langle S_1, S_2, S_3 \rangle$ where $S_1$ is a maximally consistent set of $D$-constraints over terms($\phi$) and const($\phi$); $S_2$ is a set of constraints of the form $x = d$, and $S_3$ is a set of constraints $x \equiv_K c$, where constant $K$ is the least common multiple of constants occurring in constraints $x \equiv_e y$ and $x \equiv_e y + d$.

Schüle and Schneider (Schüle and Schneider (2007)) provide a general algorithm to decide bounded $LTL(L)$ model-checking problems of infinite state systems where $L$ is a general underlying logic. An $LTL(L)$ formula $\phi$ is translated into an equivalent Büchi automaton $A_\phi$ which is symbolically represented by means of a structure defining its transition relation and acceptance condition. Then, the $LTL(L)$ model-checking problem is reduced to the $\mu$-calculus model-checking problem modulo $L$, i.e., a verification of a fixpoint formula for a given Kripke structure with respect to symbolic representations of $A_\phi$ and the underlying language $L$. Whenever properties are neither proved nor disproved over finite computations, their truth value cannot be defined. For this reason, the authors adopt a three-valued logic to evaluate formulae whose components may have undefined value. Bounded model-checking is performed essentially by computing approximate fixpoint sets of the desired formula and by checking whether the initial condition is a subset of such set of states. The work of Schüle and Schneider (2007) is based on previous results presented in Schüle and Schneider (2004), which defines a hierarchy of Büchi automata (and, therefore, temporal formulae) for which infinite state bounded model-checking is complete. The specification language of Schüle and Schneider (2004) is the quantifier-free fragment of Presburger LTL, $LTL(\text{PA})$, with past-time temporal modalities. The bounded model-checking problem is defined with respect to Kripke structures $(S, I, R)$ and it is solved by means of a reduction to
the satisfiability of Presburger formulae. In general, acceptance conditions of Büchi automata, requiring that some states are visited infinitely often, can not be handled immediately by bounded approaches which do not consider ultimately periodic models used, for instance, in the bounded model-checking approach of Biere et al. (1999) or in the encoding of Büchi automata of de Moura et al. (2002).

Therefore, Schüle and Schneider follow a different approach, tailored to bounded verification, and focus on the analysis of some classes of LTL formulae, denoted TL_F and TL_G, such that the corresponding Büchi automaton has a simpler accepting condition which does not involve infinite computations. TL_F and TL_G are the sets of LTL formulae such that each occurrence of a weak/strong temporal operator is negative/positive and positive/negative, respectively. LTL formulae are then represented symbolically by an automaton which is built using the method proposed by Clarke et al. in Clarke et al. (1994) rather than using the Vardi-Wolper construction in Vardi and Wolper (1986).

Reducing the model-checking problem to Presburger satisfiability is a rather standard approach when dealing with infinite-state systems. Demri et al. in Demri et al. (2010) show how to solve the LTL(PA) model-checking problem for the class of admissible counter systems, which are finite state automata endowed with variables over \( \mathbb{Z} \) whose transitions are labeled by Presburger formulae. In Demri et al. (2010) the authors study the decidability of the model-checking problem for admissible counter systems with respect to the first-order CTL^* language over Presburger formulae.

Hodkinson et al. study decidable fragments of first-order temporal logic in Hodkinson et al. (2000). Although some axiomatizations of first-order temporal logic are known, various incompleteness results induce the authors to study useful fragments with expressiveness between that of propositional and of first-order temporal logic. Hodkinson et al. are interested in studying the satisfiability problem and they do not consider the model-checking problem, which requires a formalism defining the interpretation of first-order variables over time. In other words, variables do not vary over time and their temporal behavior is not relevant. The languages investigated by the authors are obtained by restricting both the first-order part and the temporal part.

Bultan et al. present a symbolic model checker for analyzing programs with unbounded integer domains Bultan et al. (1999). Programs are defined by an event-action language where atomic events are expressed by Presburger formulae over programs variables \( V \). Semantics of programs is defined in terms of infinite transition systems where the states are determined by the values of variables. The specification language is a CTL-like temporal logic enriched with Presburger-definable constraints over \( V \). Solving the CTL model-checking problem involves the computation of least fixpoints over sets of programs states: the abstract interpretation of Cousot and Cousot (Cousot and Cousot, 1977) provides a method to compute approximation of fixpoints. Model-checking is done conservatively: the approximation technique admits false negatives, i.e., the solver may indicate that a property does not hold when it actually does. Programs are analyzed symbolically by means of symbolic execution techniques and they are represented by means of Presburger-definable transition systems where Presburger formulae represent symbolically the transition relation and the set of program states. Then, the state space is partitioned to reduce the complexity of verification and to obtain decidability for some classes of temporal properties, such as reachability ones. Experimental results, based on the standard Bakery algorithm and the Ticket mutual
exclusion algorithm, show the effectiveness of the method when verification involves a mutual exclusion requirement.

8 Conclusions and further developments

In this paper, we provide a procedure for deciding the satisfiability problem for CLTLB over some suitable constraint systems. The main advantage of our approach is that it allowed us to implement the first effective tool based on SMT-solvers for those logics. On one side, this method illustrates a new way to solve verification problems of formalisms dealing with variables ranging over infinite domains and having an inherent notion of discrete time as that of LTL. Instead of building an automaton for proving the satisfiability of a formula (which would be unfeasible in practice), we devise a direct method to construct one of its accepting runs which define a model for the formula. On the other hand, our framework constitutes a foundation for defining extensions to handle different temporal formalisms. In Bersani et al. (2013b) we use the same approach presented in this paper to allow for the use of variables whose behaviour is restricted to clocks into CLTLB(R, <, =). A clock is a nonnegative variable accumulating the time elapsed since the position where it was reset to 0 and that can be used to measure time between two discrete positions. When dealing with clocks, it is common to consider a uniform progression of time; the time elapsing is unique for all the clocks that are updated by the same value at each position of the discrete model. In Bersani et al. (2013b) we prove the decidability and the complexity of the satisfiability problem for the CLTLB logic endowed with a finite set of clocks, and we provide an effective implementation to solve it through SMT-solvers which extends the one presented in this work.

In Bersani et al. (2013a) we devise a reduction from MITL formulae interpreted over continuous time to CLTLB formulae with clocks. Since the reduction guarantees the equisatisfiability between the MITL formula and the resulting translation into CLTLB formulae, the satisfiability problem for the former logic can actually be solved.

References

Alur, R., Dill, D.L., 1994. A theory of timed automata. Theoretical Computer Science 126, 183–235.

Alur, R., Henzinger, T.A., 1994. A really temporal logic. Journal of the ACM 41, 181–204.

Bersani, M.M., Cavallaro, L., Frigeri, A., Pradella, M., Rossi, M., 2010. SMT-based verification of LTL specification with integer constraints and its application to runtime checking of service substitutability, in: IEEE International Conference on Software Engineering and Formal Methods, pp. 244–254.

Bersani, M.M., Frigeri, A., Morzenti, A., Pradella, M., Rossi, M., San Pietro, P., 2012. Constraint ltl satisfiability checking without automata. CoRR abs/1205.0946.
Bersani, M.M., Frigeri, A., Rossi, M., San Pietro, P., 2011. Completeness of the bounded satisfiability problem for constraint LTL, in: Reachability Problems, Springer. pp. 58–71.

Bersani, M.M., Rossi, M., San Pietro, P., 2013a. Deciding the satisfiability of mTL specifications. CoRR abs/1307.4469.

Bersani, M.M., Rossi, M., San Pietro, P., 2013b. A tool for deciding the satisfiability problem of continuous-time metric temporal logic, in: TIME 2013.

Biere, A., Cimatti, A., Clarke, E., Zhu, Y., 1999. Symbolic model checking without BDDs, in: Tools and Algorithms for the Construction and Analysis of Systems. volume 1579 of Lecture Notes in Computer Science, pp. 193–207.

Biere, A., Cimatti, A., Clarke, E.M., Strichman, O., Zhu, Y., 2003. Bounded model checking. Advances in Computers 58, 118–149.

Biere, A., Heljanko, K., Junttila, T.A., Latvala, T., Schuppan, V., 2006. Linear encodings of bounded LTL model checking. Logical Methods in Computer Science 2.

Bultan, T., Gerber, R., Pugh, W., 1999. Model-checking concurrent systems with unbounded integer variables: symbolic representations, approximations, and experimental results. ACM Transactions on Programming Languages and Systems 21, 747–789.

Clarke, E., Grumberg, O., Hamaguchi, K., 1994. Another look at LTL model checking, in: Formal Methods in System Design, Springer-Verlag. pp. 415–427.

Clarke, E., Kroening, D., Ouaknine, J., Strichman, O., 2004. Completeness and complexity of bounded model checking, in: Verification, Model Checking, and Abstract Interpretation. volume 2937 of Lecture Notes in Computer Science, pp. 85–96.

Clarke, E., McMillan, K., Campos, S., Hartonas-Garmhausen, V., 1996. Symbolic model checking, in: Computer Aided Verification. volume 1102 of Lecture Notes in Computer Science, pp. 419–422.

Comon, H., Cortier, V., 2000. Flatness is not a weakness, in: Computer Science Logic. volume 1862 of Lecture Notes in Computer Science, pp. 262–276.

Cousot, P., Cousot, R., 1977. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints, in: Proceedings of the 4th ACM SIGACT-SIGPLAN symposium on Principles of programming languages, pp. 238–252.

Demri, S., 2004. LTL over integer periodicity constraints, in: Foundations of Software Science and Computation Structures. volume 2987 of Lecture Notes in Computer Science, pp. 121–135.
Demri, S., D’Souza, D., 2002. An automata-theoretic approach to constraint LTL, in: FST TCS 2002: Foundations of Software Technology and Theoretical Computer Science. volume 2556 of Lecture Notes in Computer Science, pp. 121–132.

Demri, S., D’Souza, D., 2007. An automata-theoretic approach to constraint LTL. Information and Computation 205, 380–415.

Demri, S., Finkel, A., Goranko, V., van Drimmelen, G., 2010. Model-checking CTL* over flat Presburger counter systems. Journal of Applied Non-Classical Logics 20, 313–344.

Demri, S., Gascon, R., 2005. Verification of qualitative Z constraints, in: CONCUR 2005 - Concurrency Theory. volume 3653 of Lecture Notes in Computer Science, pp. 518–532.

Demri, S., Gascon, R., 2006. The Effects of Bounding Syntactic Resources on Presburger LTL. Technical Report LSV-06-5. LSV.

Demri, S., Gascon, R., 2007. The effects of bounding syntactic resources on Presburger LTL, in: International Symposium on Temporal Representation and Reasoning (TIME), IEEE Computer Society. pp. 94–104.

Hodkinson, I.M., Wolter, F., Zakharyaschev, M., 2000. Decidable fragment of first-order temporal logics. Annals of Pure and Applied Logic 106, 85–134.

Holzmann, G., 1997. The model checker SPIN. IEEE Transactions on Software Engineering 23, 279 –295.

Kamp, J.A.W., 1968. Tense Logic and the Theory of Linear Order. Ph.D. thesis. University of California at Los Angeles.

Kroening, D., Strichman, O., 2003. Efficient computation of recurrence diameters, in: Verification, Model Checking, and Abstract Interpretation. volume 2575 of Lecture Notes in Computer Science, pp. 298–309.

Microsoft Research, 2009. Z3: An efficient SMT solver. http://research.microsoft.com/en-us/um/redmond/projects/z3/.

de Moura, L.M., Rueß, H., Sorea, M., 2002. Lazy theorem proving for bounded model checking over infinite domains, in: Automated Deduction-CADE-18. volume 2392 of Lecture Notes in Computer Science, pp. 438–455.

Pradella, M., Morzenti, A., San Pietro, P., 2013. Bounded satisfiability checking of metric temporal logic specifications. ACM Transactions on Software Engineering and Methodology (TOSEM) 22.

Safra, S., 1988. On the complexity of omega-automata, in: IEEE Annual Symposium on Foundations of Computer Science, IEEE Computer Society. pp. 319–327.
Schüle, T., Schneider, K., 2004. Bounded model checking of infinite state systems: exploiting the automata hierarchy, in: Proceedings of the ACM and IEEE International Conference on Formal Methods and Models for Co-Design, pp. 17 – 26.

Schüle, T., Schneider, K., 2007. Bounded model checking of infinite state systems. Formal Methods in System Design 30, 51–81.

Sistla, A.P., Clarke, E.M., 1985. The complexity of propositional linear temporal logics. Journal of the ACM 32, 733–749.

Vardi, M.Y., Wolper, P., 1986. An automata-theoretic approach to automatic program verification, in: Proceedings, Symposium on Logic in Computer Science, IEEE Computer Society. pp. 332–344.