Counter Systems with Presburger-definable Reachability Sets : Decidability and Complexity

Amit Kumar Dhar
Master Parisien de Recherche en Informatique
work done at LSV, ENS Cachan
under the guidance of
Stéphane Demri
LSV, ENS Cachan
Arnaud Sangnier
LIAFA, Paris 7

18^{th} August 2011

General Context

Model checking deals with the techniques of verifying whether a given formula in a suitably expressive logic is satisfied in a given abstract structure. The techniques as well as the cost for such task vary depending on the abstract formalism used to represent the model and the logic used to express the properties [CGP00]. For abstract formalism, it is worth to note that most of the practical systems are infinite-state systems. The challenge is to manipulate infinite sets of configurations. Whereas many abstract formalisms for representing infinite-state system exist, most of them tend to have an undecidable model checking. Counter systems are such a formalism used in many places like, broadcast protocol [EFM99], programs with pointers [BFLS06], and data words (XML) [BMS$^+$06], [DL06]. But, alongwith their large scope of usability, many problems on general counter systems are known to be undecidable. For example, systems with two counters are known to simulate Turing Machine operations [Min67]. We consider the Linear Temporal Logic for specifying properties. LTL is enough powerful to specify temporal properties like liveness or safety conditions and yet have many desirable properties. We consider here LTL with the basic Until and Next operator. Later we also look into basic LTL extended with linear constraints on counter as atomic propositions.

The Problem Studied

A more general kind of counter systems, namely "flat counter System with finite monoid property" was first studied in [FL02] where the reachability problem for these systems was shown to be decidable. Later, in [DFGvD10], it was shown that CTL* is decidable for specific kind of counter systems called "flat Admissible Counter System (ACS)", by reduction to Presburger Arithmetic, the decidable first-order logic of natural numbers with comparisons and addition. Whereas CTL* can express richer

[1]The report is written in English, because the author knows very little French

properties than reachability, the procedures described there have high complexity upto 4-EXPTIME. Our goal is to study the lower bounds of subclasses of this kind of systems and investigate tight upper bounds for model checking LTL and its fragments.

Proposed Contribution

Here, we obtain a tight complexity bound on LTL model checking over such system and show that the problem is in fact NP-complete. For this, we first show the NP-completeness of model checking LTL on flat Kripke structures using "general stuttering principle" [KS05]. Next, using this result, we show that LTL model checking on counter systems with linear sums as updates is NP-complete. For this result we also use the "small solution property" [BT76] of system of equations. The small solution property by itself is sufficient to decide the reachability problem on counter system and have been used to show decidability of reachability problem of different type of systems in [Rac78],[GI81]. Later building on the previous results we show the NP-completeness of checking LTL with counter constraint over counter systems with updates as linear sums.

Future Work and Perspectives

Due to the relative lower complexity of the algorithms developed, I plan to implement the algorithms developed or a variant of that is well-suited for implementation. There are several related problems which are not addressed here, like: extension of the logic with past operators, model-checking for CTL*, model checking over flat counter systems with no restriction over the guards, other extensions of counter systems like Affine counter system with finite monoid property [FL02] etc. Though as stated earlier, all these problems are known to be decidable but the upper bounds known for these problems are very crude. Hence it is interesting to investigate the problems to look for tighter upper bounds. Finally, the restrictions on the counter systems may as well be increased to find out when the model checking problem falls below NP.

1 Introduction

Model checking counter systems Model Checking is a well-known approach in computer science to verify properties of computing systems, in order to say that a given system always does something good or it never does something bad. For this purpose, the computing system is represented in abstract structures leaving out irrelevant details and the property that is to be verified is expressed by a formula in a suitable logic, thus reducing the problem to checking the satisfaction of a given formula by a given model. Model checking deals with the techniques of verifying whether such a given formula in a suitably expressive logic is satisfied in a given abstract structure. The techniques as well as the cost for such task varies depending on the abstract formalism used to represent the model and the logic used to express the properties [CGP00]. For abstract formalism, it is worth to note that most of the practical systems are infinite-state systems. The challenge is to manipulate infinite sets of configurations. Whereas many abstract formalisms for representing infinite-state system exist, most of them tend to have an undecidable model checking. Counter systems are such a formalism used in many places like, broadcast protocol [EFM99], programs with pointers [BFLS06], and data words (XML) [BMS$^+$06], [DL06]. But, alongwith their large scope of usability, many problems on general counter systems are known to be undecidable. For example, systems with two counters are known to simulate Turing Machine operations [Min67]. However, certain restricted classes of counter systems have decidable properties. Restriction on counter systems may be imposed on counter operations, guards, underlying structure of the system (e.g. flatness), boundedness of counter values (e.g. reversal-boundedness) etc. Here, we concentrate on flat counter systems, where the restriction is imposed on the underlying structure. For the logic used for specifying properties, again there are many logics and subclasses, like LTL, CTL, CTL*. We consider the Linear Temporal Logic for specifying properties. LTL is enough powerful to specify temporal properties like liveness or safety conditions and yet have many desirable properties. We consider here LTL with the basic Until and Next operator. Later we also look into basic LTL extended with linear constraints on counter as atomic propositions.

Motivations A more general kind of counter systems, namely "flat counter System with finite monoid property" was first studied in [FL02] where the reachability problem for these systems was shown to be decidable. Later, in [DFGvD10], it was shown that CTL* is decidable for specific kind of counter systems called "flat Admissible Counter System (ACS)", by reduction to Presburger Arithmetic, the decidable first-order logic of natural numbers with comparisons and addition. Whereas CTL* can express richer properties than reachability, the procedures described there have high complexity upto 4-EXPTIME. Our goal is to study the lower bounds of subclasses of this kind of systems and investigate tight upper bounds for model checking LTL and its fragments.

Goal and results We start with the goal of finding optimal complexity of model checking LTL with counter constraints over counter systems. But, counter systems in its full generality is known to be Turing complete, and thus have undecidable reachability. So, we look into decidable fragments of counter systems which restricts the counter systems structurally and also in the type of updates that are used. The type of counter systems we explore are flat and have only linear sum over counter values. These type of systems are a subclass of the kind of system shown to have decidable CTL* model checking. Here, we obtain a tight complexity bound on LTL model checking over such system and show that the problem is in fact NP-complete. For this, we first show the NP-completeness of model checking

LTL on flat Kripke structures using "general stuttering principle" [KS05]. Next, using this result, we show that LTL model checking on counter systems with linear sums as updates is NP-complete. For this result we also use the "small solution property" [BT76] of system of equations. The small solution property by itself is sufficient to decide the reachability problem on counter system and have been used to show decidability of reachability problem of different type of systems in [Rac78],[GI81]. Later building on the previous results we show the NP-completeness of checking LTL with counter constraint over counter systems with updates as linear sums.

Structure In Section 3, we start with the preliminary definitions of logics and considered systems. Then, in Section 4, the problem of model checking LTL over flat Kripke structures. The result presented here is an adaption of the general stuttering principle to a restricted case. In Section 5, we extend the model with counters and consider flat counter systems and model checking simple LTL over them. Later in Section 6, we consider a richer problem of investigating the complexity of model checking LTL formulas with linear counter constraints as atomic propositions over flat counter systems. In Section 7, we present possible extensions and open problems related to the problems studied here. Due to constraint of space, omitted proofs can be found in the technical appendix.

2 Related Work

Counter systems are a very popular class of infinite-state systems studied and analyzed. Fragments of counter systems over which some problems like reachability are decidable, have been studied extensively. For example, counter systems for which the reachability set is effectively semilinear (or equivalently the reachability relation is effectively semilinear) is studied in [CJ98]. Again a more general class of counter systems with decidable reachability property is studied in [ISD$^+$00]. Though, restrictions are placed on counter systems to make them decidable, there are some classes of counter systems like the *flat admissible counter system with finite monoid property* which have decidable reachability [FL02] and also decidable CTL* model checking [DFGvD10]. While optimal bounds of model checking simple LTL on Flat Kripke structure are studied earlier in [Kuh10] and [KF11], but tight bound for model checking simple LTL or LTL with counter constraints on counter systems were still open.

3 Preliminaries

In this section, we provide the definitions of structures used throughout the thesis. At the end we define the background of the problem and the problem itself that we want to study.

3.1 Counter Systems

For abstract models we use flat counter systems with linear sum on counters as constraints and its fragments.

Definition 1. *A counter system of dimension n is a labelled graph $\mathcal{C} = \langle Q, \delta, \Sigma, C, q_{init} \rangle$ consisting of*

- *n counters represented as $C = (c_1, \cdots, c_n)$*

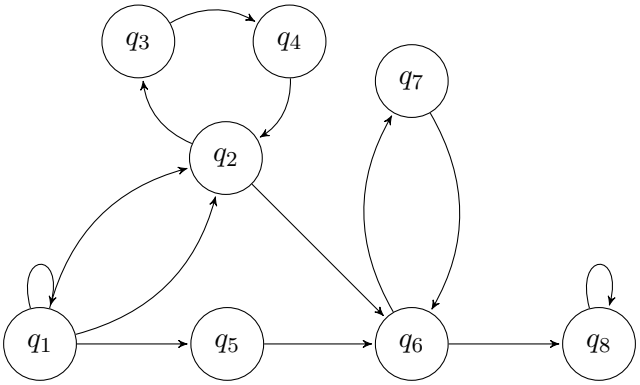


Figure 1: A flat-counter system

- Σ *is a finite set of tuples of the form* $(g, \overrightarrow{u})$*, where* g *is used to check the counter values and* $\overrightarrow{u} \in \mathbb{Z}^n$ *is a vector to update the counter values.*
  g *is a conjunction of conditions of the form* $\sum_i a_i c_i \sim k$ *for* $1 \leq i \leq n$*, where* c_i *is a counter whose value is compared with any constant* $k \in \mathbb{Z}$*,* $a_i \in \mathbb{Z}$ *are any constant and* $\sim \in \{=, \leq, \geq, <, >\}$*.* $\overrightarrow{u}$ *is a vector whose each element* $\overrightarrow{u}[i] \in \mathbb{Z}$*, where* $\overrightarrow{u}[i]$ *denotes the* i^{th} *element of* $\overrightarrow{u}$*.*

- Q *is a finite set of states.*

- $\delta \subseteq Q \times \Sigma \times Q$ *is a transition relation. For each transition* $t = q \xrightarrow{(g, \overrightarrow{u})} q'$*, we denote by* $source(t)$ *and* $target(t)$ *as the source and target states of each transition. Similarly,* $guard(t)$ *and* $update(t)$ *denotes the guard* g*, and update* $\overrightarrow{u}$*, of the transition* t*.*

- q_{init} *is the initial state.*

Every counter system $\mathcal{C}$ with n counters naturally induces a transition system $T_{\mathcal{C}} = \langle S, \rightarrow \rangle$ where $S = Q \times \mathbb{N}^n$ is the set of configurations and $\rightarrow \subseteq S \times \delta \times S$. Also, $t = \langle q, \overrightarrow{x} \rangle \xrightarrow{(g, \overrightarrow{u})} \langle q', \overrightarrow{x'} \rangle$ (also denoted as $\langle q, \overrightarrow{x} \rangle, t, \langle q', \overrightarrow{x'} \rangle$) iff

- $q = source(t)$
- $q' = target(t)$
- $\overrightarrow{x}$ satisfies guard(t)
- $\overrightarrow{x'} = \overrightarrow{x} + update(t)$

where $\overrightarrow{x'}$, $\overrightarrow{x}$, $\overrightarrow{u}$ are elements of $\mathbb{N}^n$. For a finite word $w \in \delta^+$, $w = t_0 t_1 \cdots t_k$, we have $\langle q, \overrightarrow{x} \rangle \xrightarrow{w} \langle q', \overrightarrow{x'} \rangle$ if $\exists \langle q_0, \overrightarrow{y_0} \rangle, \cdots, \langle q_{k+1}, \overrightarrow{y_{k+1}} \rangle \in S$ such that $\langle q_0, \overrightarrow{y_0} \rangle = \langle q, \overrightarrow{x} \rangle$ and $\langle q_k, \overrightarrow{y_k} \rangle = \langle q', \overrightarrow{x'} \rangle$ and for $0 \leq i \leq k$, $\langle q_i, \overrightarrow{y_i} \rangle \xrightarrow{t_i} \langle q_{i+1}, \overrightarrow{y_{i+1}} \rangle$.

An infinite word $w \in \delta^\omega$ is fireable if for all finite prefix w' of w, there exists $\langle q, \overrightarrow{x} \rangle \in S$ such that $\langle q_{init}, \overrightarrow{0} \rangle \xrightarrow{w'} \langle q, \overrightarrow{x} \rangle$. In other words, $w = t_0 t_1 \cdots$ is fireable iff there exists an infinite sequence of

configurations $\langle q_i, \overrightarrow{x_i}\rangle$ for $i \in \mathbb{N}$ such that $\forall i, \langle q_i, \overrightarrow{x_i}\rangle \xrightarrow{t_i} \langle q_{i+1}, \overrightarrow{x_{i+1}}\rangle$ and $\langle q_0, \overrightarrow{x_0}\rangle = \langle q_{init}, \overrightarrow{0}\rangle$. A *run* ρ in $\mathcal{C}$, is an infinite path in $T_{\mathcal{C}}$ denoted as

$$\rho = (q_0, \overrightarrow{x_0}) \xrightarrow{t_0} \cdots \xrightarrow{t_{k-1}} (q_k, \overrightarrow{x_k}) \xrightarrow{t_k} \cdots$$

we denote by $\delta(\rho)$ the infinite word $w = t_0 t_1 \cdots t_k \cdots$. Also, we denote by $fireable(\mathcal{C})$, the set of infinite words in δ^ω which are fireable.

A *flat counter system* is a counter system in which every node in the underlying graph is part of at most one cycle. In spite of this restriction, flat counter systems are found in many places in practice [LS05]. Also, despite their simplicity of structures, not all flat counter system have decidable model checking problem [Cor02]. Here, we look at flat counter systems with *finite monoid property*, for which decidability was proved in [DFGvD10]

The system obtained by leaving out counter operations and guards from a counter system is also called a *Kripke structure*. Alternative, a Kripke structure can be viewed as a counter system with $C = \{c_1\}$, all guards are $True$ and all update $u \in \mathbb{Z}$ equal to zero vector. In a Kripke structure every node is labelled by proposition symbols. Without loss of generality, we may assume that each state satisfies a unique proposition. A *control path* π in a counter system/Kripke structure is defined as a sequence of nodes from the structure which forms an infinite path in the underlying graph of the structure.

$$\pi = q_0, q_1, \cdots, q_k, \cdots$$

where, every $q_i \in Q$. For counter systems, a control path π, can also be seen as the projection of states from the corresponding run ρ. A control path is also considered a word over states, i.e. $\pi \in Q^\omega$. For any such control path π, we denote by π_i as the path obtained from π by deleting the first i states and by $\pi(i, j)$ the subpath $(\pi(i), \pi(i+1), \cdots, \pi(j))$ where $\pi(i)$ denotes the i^{th} node in $\pi(i)$. We also extend the definition of the function δ defined on run, for the case of the control paths similarly as, $\delta(\pi)$ is the sequence of transitions taken for the control path π.

3.2 Temporal Logic

For model checking, we consider LTL formula whose atomic predicates are propositional symbols and linear constraints on counters of the form $\sum_{1 \le i \le n} a_i c_i \sim k$, where c_i is a counter whose value is compared with any constant $k \in \mathbb{Z}$, $a_i \in \mathbb{Z}$ are any constant and $\sim \in \{=, \le, \ge, <, >\}$. Also, the logic is equipped with U and X as the only temporal operators. The formulas are defined as follows:

$$\varphi = \top \mid p \mid \textstyle\sum_i a_i c_i \sim k \mid \neg\varphi \mid \varphi \wedge \varphi \mid X\varphi \mid \varphi U \varphi$$

where p is propositon symbol, which in our case are state symbols. We call this class of formulas as $LTL_C(U, X)$. Thus, models of $LTL_C(U, X)$ are elements of $(Q \times \mathbb{N}^n)^\omega$. Given a run $\rho = (q_0, \overrightarrow{x_0}) \xrightarrow{t_0} \cdots \xrightarrow{t_{k-1}} (q_k, \overrightarrow{x_k}) \xrightarrow{t_k} \cdots$, the satisfaction relation $\models$ is defined over a run ρ as:

- $\rho, i \models \top$. $\top$ is taken as $True$ and is satisfied by all runs.
- $\rho, i \models p$ iff $p = q_i$.
- $\rho, i \models \sum_i a_i c_i \sim k$ iff $\overrightarrow{x_i} \models \sum_j a_j c_j \sim k$ using the valuation that $c_j = \overrightarrow{x_i}[j]$

- $\rho, i \vDash \neg\phi$ iff $\rho, i \nvDash \phi$.
- $\rho, i \vDash \phi \wedge \psi$ iff $\rho, i \vDash \phi$ and $\rho, i \vDash \psi$.
- $\rho, i \vDash X\phi$ iff $\rho, i+1 \vDash \phi$.
- $\rho, i \vDash \phi U \psi$ iff there exist $j \geq 0$ such that $\rho, j \vDash \psi$ and for all $i \leq k < j$, $\rho, i \vDash \phi$.

In general, for a run ρ and a formula φ we say, $\rho \vDash \varphi$ iff $\rho, 0 \vDash \varphi$. $LTL(U, X)$ is defined in the same way as $LTL_C(U, X)$ with all the counter constraints removed, hence having only propositional symbols as atomic propositions. Also, models of $LTL(U, X)$ are restricted to elements of Q^ω, for the case where Kripke structures are considered. Formulas from $LTL(U, X)$ are interpreted over a control path. For example the formula

$$\top U(q_7 \wedge Xq_6)$$

says that finally there is a node which is labelled by q_7 and the next position is labelled by q_6 and is satisfied by the control path $q_1q_5q_6q_7q_6q_8q_8\cdots$ among others in the counter system of Figure 1. We define the *temporal depth* of any LTL formula φ, written as $td(\varphi)$, as the maximum nesting level of the temporal operators occuring in φ. For example the LTL formula presented above has temporal depth 2 as X occurs within U.

3.3 Paths, Schemas and Runs

For a given flat counter system $\mathcal{C} = \langle Q, \delta, \Sigma, C\rangle$, a *path segment* σ, is defined as a finite sequence of transitions $t_0t_1\cdots t_k$ with $k \geq 0$ i.e. $\sigma \in \delta^*$ as defined earlier. We denote by $\sigma(i)$ the i^{th} transition in σ. A path segment $t_0t_1\cdots t_k$ is simple if $k > 0$ and for all $0 \leq i, j \leq k$ $i \neq j$ we have that $t_i \neq t_j$. A *loop* is defined as a simple path segment σ for which $source(t_0) = target(t_k)$. Otherwise it is called a *non-loop segment*. For a path segment σ, $\sigma_{\leq i}$ denotes $(t_0, t_1, \cdots, t_i)$, a prefix path segment of length $i+1$ for $0 \leq i < |\sigma|$. A *path schema*, $L = \langle y_0, l_0, y_1, \cdots, y_k, l_k\rangle$ is defined as a sequence of path segments following an expression such that:

- For all $i \in [1, k]$, y_i is a simple non-loop segment.
- For all $i \in [0, k]$, l_i is a loop.
- y_0 is a non-loop segment.
- For all $i \in [1, k]$ $target(l_{i-1}(|l_{i-1}|)) = source(y_i(0))$ and $target(y_i(|y_i|)) = source(l_i(0))$. If y_0 is non-empty, then $target(y_0(|y_0|)) = source(l_1(0))$.
- For any transition t occuring in any segment y_i, t does not occur in any of the segments l_j for $j > i$
- $\sum_i update(l_k(i)) \geq 0$
- For a flat counter system $\mathcal{C}$, if L is a path schema then for $i \neq j$, we have $y_i \neq y_j$ and $l_i \neq l_j$.

For a path schema L, $S_L = \{y_0, l_0, y_1, l_1 \cdots, y_k, l_k\}$ denotes the set of path segments in the schema. Since we consider only flat counter systems, there is a natural ordering of the path segments in a path schema. Also, $L_{<\sigma}$ denotes the path schema obtained from L by taking all path segments which are strictly before σ in L in ordering. For example, $L_{<l_i} = \langle y_0, l_0, y_1, l_1 \cdots, y_i\rangle$. The *language* associated with L is a collection of ω-words, which can be represented using the ω-regular expression $\mathcal{L}_L = y_0 l_0^+ y_1 \cdots y_k l_k^\omega$. We also extend the definition of $fireable$ sequence of transitions to L in a counter system $\mathcal{C}$ as $fireable(L) = \mathcal{L}_L \cap fireable(\mathcal{C})$. For a word $w \in fireable(L)$, thus we can associate a unique vector of integer denoted as $\overrightarrow{M}(w) \in (\mathbb{N} \setminus \{0\})^k$ such that $w = y_0 l_0^{\overrightarrow{M}(w)[0]} y_1 \cdots y_k l_k^\omega$, where $\overrightarrow{M}(w)[i]$ is the i^{th} element of $\overrightarrow{M}(w)$ for $0 \leq i \leq k-1$. In the same way for a run ρ such that $\delta(\rho) \in fireable(L)$, we define $\overrightarrow{M}(\rho) = \overrightarrow{M}(\delta(\rho))$. Also, $realizable(L)$ is a subset of vectors of integers $realizable(L) \subseteq (\mathbb{N} \setminus \{0\})^k$, $k+1$ being the number of loops in L, such that $realizable(L) = \{\overrightarrow{M}(w) | w \in fireable(L)\}$. Clearly, the maximum possible size of any path schema, $|\delta|$, is bounded by the size of the structure.

Lemma 1. *The total number of path schemata in a flat counter system is finite and is at most exponential in the size of the system.*

Proof. We know that the size of a simple path segment is bounded by $|\delta|$. Hence, the number of possible simple path segments is bounded by $|\delta|^{|\delta|}$. Also, the number of loops are bounded by $|\delta|$. Since, a path schema contains loops and simple path segments alternatively, the number of possible path schema is bounded by $(|\delta|^{|\delta|}.|\delta|)^{|\delta|}$. □

We define the model checking problem MC(L,CS), where L is a fragment of $LTL_C(U, X)$ (possibly equal to it) and CS is a class of counter systems (possibly the whole class) as following:

Input: A system $\mathcal{C} \in$CS , a configuration $(q, \overrightarrow{x})$ and a formula $\phi \in L$

Output: Is there a run ρ starting with $(q, \overrightarrow{x})$ in $\mathcal{C}$ such that $\rho \vDash \phi$?

This is a the existensial version of the model checking problem. Note that without loss of generality, we can assume that the initial value of all counters in a counter system is 0, as we can always put transitions as a prefix to the run to reach any given counter value. We also consider many fragments of counter systems like:

- PS - path schema. Note that path schema is not exactly a counter system in the way that, each loop in a path schema is assured to be taken at least once, which is not the case with counter system.
- PS_n - path schema with n loops, for a fixed $n \in \mathbb{N}$,
- FKS - flat Kripke structures,
- FCS - flat counter systems.

The main goal is to provide an optimal algorithm for MC($LTL_C(U, X)$,FCS). This problem can be proved decidable as corollary of the following proposition

Proposition 1. *[DFGvD10]* MC($LTL_C(U, X)$, FCS) *is decidable.*

It is also shown that the decidability holds even when the logic can be extended to CTL* with past-time operator and with first order quantification over counter values. But the complexity of the presented procedure is very high and lower bounds are also unknown. We here obtain optimal complexity bound by combining two well-known methods: General stuttering principle [KS05] and small solutions to equations [BT76] leading to short runs [Rac78]. The procedure proceeds by transforming the formula from $LTL_C(U,X)$ to one in $LTL(U,X)$ and also transforming the counter system to preserve satisfiability. Then, we characterize the valid runs in the structure using system of equations. Then, a polynomial size solution to the equations is guessed and checked which is guaranteed by the two stated methods of "General stuttering principle" and "small solutions to equations". A solution to the equations gives us a run in the system satisfying the formula.

4 Model Checking on Path Schema

In this section, we prove that MC($LTL(U,X)$,PS) is NP-complete, where PS denotes the class of path schema. As a corollary, we obtain the same complexity characterization on flat Kripke structures. Moreover, it is shown in Theorem 3, that further restriction of the model to fixed number of loops, reduces the complexity of the problem to polynomial time. We will first prove the following:

Theorem 1. *Given a path schema $L = \langle y_0, l_0, y_1, l_1 \cdots l_k \rangle$, and a formula $\varphi \in LTL(U,X)$, if there exists a word $w = y_0 l_0^{i_0} y_1 \cdots l_{k-1}^{i_{k-1}} y_{k-1} l_k^{\omega}$ such that for a run ρ with $\delta(\rho) = w$ and $\rho \vDash \varphi$ then there exists a word $w' = y_0 l_0^{i'_0} y_1 \cdots l_{k-1}^{i'_{k-1}} y_{k-1} l_k^{\omega}$ and a run ρ' with $\delta(\rho') = w'$ and $\rho' \vDash \varphi$ and for all $0 \leq j < k$, $i'_j \leq 2.td(\varphi) + 2$, where $td(\varphi)$ denotes the temporal depth of φ as defined above.*

To get a tight bound on the number of unfoldings, we consider the depth of U and X separately. The maximum depth of the temporal operator U (respectively X) in a formula is the maximum nesting depth of U (respectively X) in the formula. We write $LTL(U^m, X^n)$ to denote the set of formulas in $LTL(U,X)$ with maximum depth of U and X operator, m and n respectively. Consider an infinite word of transition $w = t_0 t_1 t_2 \cdots$ such that w belongs to $fireable(L)$ for a given path schema L. A finite word of transitions $u \in \delta^+$ is (m,n)-*redundant* in w if there exists a finite word of transitions (possibly empty) w_1 and an infinite word of transitions w_2 such that:

$$w = w_1.u^k.w_2$$

where $k.|u| \geq (m+1).|u| + n + 1$ (that is in w the word u is repeated enough number of times to cover $(m+1).|u| + n + 1$ letters). For a run/control path ρ in a system without counters, such that $\delta(\rho) \in \mathcal{L}_L$, u *starts* for $\rho(i)$ is defined as $source(u(0)) = \rho(i)$.

For a word of transitions u, which is (m,n)-redundant in w, we fix the functions $begin(w,u) \in \delta^+$, $end(w,u) \in \delta^\omega$ and $iter(w,u) \in \mathbb{N}$ (with $iter(w,u) \geq (m+1).|u| + n + 1$) such that $w = begin(w,u).u^{iter(w,u)}.end(w,u)$. We define $w \setminus u = begin(w,u).u^{iter(w,u)-1}.end(w,u)$. Note that there might be different choices for the function *begin*, *end* and *iter*, since there might be more than one way of partitioning the word, but for our case one of the partitions is chosen arbitrarily. We also define the relation $\rightarrow_{m,n}$ as: two runs α, β are said to be related as $\alpha \rightarrow_{m,n} \beta$ if β is obtained from α by removing the first iteration of an (m,n) redundant loop, u, that is, $\delta(\beta) = \delta(\alpha) \setminus u$. For a finite word of transitions $u \in \delta^+$ such that $u = t_0 t_1 \cdots t_l$, for $0 \leq k < |u|$, $u[k]$ is the word $t_{k.mod|u|}.t_{(1+k).mod|u|} \cdots t_{(l+k).mod|u|}$. We will prove the following claim :

Claim 1. *For two runs α and β, if $\alpha \rightarrow_{m,n} \beta$, and for $\varphi \in LTL(U^m, X^n)$ $\alpha \models \varphi$ iff $\beta \models \varphi$.*

It is evident that, if the claim is proved then for any formula satisfied by a run in the given structure can be satisfied by a run with bounded numbers of iteration of each loop (possibly by a series of removal of first iterations of the loops that occur more than the specified number of times in the run). Also, note that though the quantity $m + 1 + \left\lceil \frac{n+1}{|u|} \right\rceil$ is not exactly $td(\varphi)$, but for $|u| \geq 1$ we have, $m + 1 + \left\lceil \frac{n+1}{|u|} \right\rceil \leq 2.td(\varphi) + 2$. Hence this proves Theorem 1.

The proof of the Claim can be found in the technical appendix.

The proof of the Claim as presented, though in the line of the proof of *the general stuttering principle* [KS05], is suited for path schemas. The proof presented in [KS05] deals with Kripke structures which are non-flat and also gives a tighter bound on the number of iterations. For our case, we have simplified the proof for flat Kripke structures and also for less tight upperbound as needed by later proofs. This helps in extending the proof and applying it to more general models as presented in the later sections.

As a result of the previous theorem, we can state the following lemma,

Lemma 2. MC($LTL(U, X)$,PS) *is in* NP*.*

The proof proceeds by guessing the number of times each loop is taken and then checking the formula on a constructed *uniformly periodic path* in polynomial time. The detailed proof can be found in the technical appendix.

Lemma 3. MC($LTL(U, X)$,PS)*, is* NP*-hard.*

The proof proceeds by a reduction of the satisfiability problem of propositional logic to the given problem. Details about the reduction can be found in the technical appendix.

From the above lemmas, we have the following:

Theorem 2. MC($LTL(U, X)$, PS) *is* NP*-complete.*

From the previous proof of NP-completeness, we can say the following:

Corollary 1. *[KF11]* MC($LTL(U, X)$, FKS)*, is* NP*-complete, where FKS denotes flat Kripke structure.*

This follows directly from the above proof, by additionally guessing the path schema in the structure which is of size polynomial in the size of the structure and then checking the formula on the path-schema. In the NP-hardness proof in [KF11], the non-determinism is simulated from the non-determinism in structure instead of the number of loops taken as in this case.

Furthermore, we can also restrict the number of loops in a path schema. But in this case the complexity is much lower.

Theorem 3. MC($LTL(U, X)$, PS_n) *is in* P *where PS_n denotes path schemas with at most n loops for some fixed $n \in \mathbb{N}$.*

The proof can be found in the technical appendix.

Note that from the NP-hardness proof of [KF11], where the proof deals with a flat Kripke structure with a unique loop, we have the following:

Corollary 2. *[KF11]* MC($LTL(U,X)$, FKS_n)*, is* NP*-complete, where* FKS_n *denotes flat Kripke structure with number of loops restricted to be at most some fixed constant,* n*.*

So, to summarize, we obtain that model checking $LTL(U,X)$ over path schemas is NP-complete. Whereas if we restrict the model to be path schemas with a fixed number of loops, then the model checking problem of the same kind of formula is in P. Also, we show that, MC($LTL(U,X)$, FKS) is NP-complete where FKS denotes flat Kripke structure. As a summary,

- MC($LTL(U,X)$,PS) is NP-complete.
- MC($LTL(U,X)$,FKS) is NP-complete.
- MC($LTL(U,X)$,FKS_n) is NP-complete.
- MC($LTL(U,X)$,PS_n) is P.

5 Model Checking Simple LTL over Flat Counter Systems

Now, let us consider a more general and powerful system than Kripke structures namely flat counter system in FCS. A flat counter system is represented as $\mathcal{C} = \langle Q, \delta, \Sigma, C, q_{init} \rangle$ as defined earlier and we will look into the complexity of model checking $LTL(U,X)$ on such systems. We shall prove the following:

Theorem 4. MC($LTL(U,X)$,FCS) *is* NP*-complete where* FCS *denotes the class of flat counter systems.*

Intuitively, we will formulate a system of equations characterizing the infinite runs over the structure which are valid with respect to the counter values. The equations contain as variables the number of times each loop is taken. We use the small solution property of integer equations, to guess a polynomial size solution of the equations. Though this is sufficient for deciding reachability, it is not sufficient for deciding $LTL(U,X)$. Hence, we use Theorem 1, to guess a path satisfying the formula and then we compare the two solutions to decide whether the formula is satisfiable.

5.1 Characterizing infinite runs

Here, we will define the system of equations which characterizes the set of valid infinite runs of a path schema. That is, the equations will characterize the set $realizable(L)$. We have variables identifying how many times each loop is taken. The equations ensures that any path satisfying all the equations are valid, i.e. they respect the guards and the updates. This is achieved by expressing the effect of updates in the transitions taken, on the counters and then checking them with the guards in the system at appropriate places.

Consider a path schema $L = \langle y_0, l_0, y_1, l_1 \cdots, y_k, l_k \rangle$. For a path segment $\sigma \in S_L$ (which is a word in δ^+), we define the effect of the path segment on a counter c_i as,

$$\triangle_{=\sigma}(i) \stackrel{def}{=} \sum_{0 \leq j \leq |\sigma|} update(\sigma(j))[i]$$

We extend the definition of $\triangle_{=\sigma}$ to path schema. But in this case, due to the involvemnent of loops and the number of times they are taken in the effect, we obtain an expression for the effect of a path schema on a counter c_i with variables in it. In the expressions, X_σ is a variable signifying how many times the path segment σ is taken. The expressions for effect are obtained as described below,

$$\triangle_{<\sigma}(i) \stackrel{def}{=} \sum_{\sigma' \in S_{L_{<\sigma}}} X_{\sigma'}.\triangle_{=\sigma'}(i)$$

The expression is a sum of the effects of all the path segments σ' in the path schema taken $X_{\sigma'}$ times which occur strictly before σ in the path schema. Since, all y_is in a path schema are taken only once in any run through the path schema, we also impose the condition that X_{y_i} for all $0 \leq i \leq k$ is equal to 1. Also, we emphasize that each of the segment is taken at least once. Hence $X_\sigma \geq 1$ for any σ.

Now, we will define a system of equations which characterizes the valid infinite runs ρ such that $\delta(\rho) \in fireable(L)$. The set of equations for characterizing valid runs, $\mathcal{E}$, is the smallest set of equations which contains the following type of equations:

1. For $i \in [0, k]$, we add the following equation.

   $$X_{y_i} = 1$$

   This ensures that each of the non-loop path segment is taken only once.

2. For $i \in [0, k-1]$, we add the following equation.

   $$X_{y_i} \geq 1$$

   This ensures that each of the loops is taken at least once.

3. For each $\sigma \in S_L$ and for all $0 \leq j < |\sigma|$, for all constraints of the form $\sum_i a_i c_i \sim k$ in $guard(\sigma(j))$, $\mathcal{E}$ contains an equation of the form:

   $$\sum_{1 \leq i \leq n} a_i.(\triangle_{<\sigma}(i) + \triangle_{=\sigma_{\leq j-1}}(i)) \sim k$$

   This ensures that each segment can be taken at least once. In other words, the equation ensures that each of the guards on any transition is satisfied by the counter values upto that guard in a given configuration path.

4. For each $\sigma \in S_L \setminus \{l_k\}$ (except the last loop) and for all $0 \leq j \leq |\sigma|$, for all constraints of the form $\sum_i a_i c_i \sim k$ in $guard(\sigma(i))$, $\mathcal{E}$ contains an equation of the form:

   $$\sum_{1 \leq i \leq n} a_i.(\triangle_{<\sigma}(i) + (X_\sigma - 1).\triangle_{=\sigma}(i) + \triangle_{=\sigma_{\leq j-1}}(i)) \sim k$$

   This type of equations, effectively only for loops, ensures that in the last iteration of a loop too, the guards on the transitions of the loop are satisfied by the counter values upto that transition in the configuration path.

5. For each counter c_i and for each $\sigma \in S_L \setminus \{l_k\}$ and for all $0 \leq j \leq |\sigma|$, $\mathcal{E}$ contains equations of the form:

$$\triangle_{<\sigma}(i) + \triangle_{=\sigma_{\leq j-1}}(i) \geq 0$$
$$\triangle_{<\sigma}(i) + (X_\sigma - 1).\triangle_{=\sigma}(i) + \triangle_{=\sigma_{\leq j-1}}(i) \geq 0$$

And for each counter c_i and $\sigma = l_k$ and for all $0 \leq j \leq |\sigma|$, $\mathcal{E}$ contains equations of the form:

$$\triangle_{<\sigma}(i) + \triangle_{=\sigma_{\leq j-1}}(i) \geq 0$$

The effect on each of the counters till the transition in the configuration path is positive, similarly at the starting and ending iteration of repetition for loops. Note that, since we don't have any variable defined for the last loop, we don't have a formula of the second type for the last loop.

6. For the last loop l_k, and for all $0 \leq j \leq |l_k|$, $\mathcal{E}$ contains an equation for $guard(l_k(j))$ which are obtained by replacing each constraint of $guard(l_k(j))$ of the form $\sum_i a_i c_i \sim k$ where $\sim \in \{<, \leq, =\}$ by a formula as specified below:

$$\sum_i a_i \triangle_{=l_k}(i) = 0$$

For $\sim \in \{<, \leq, =\}$ in any constraint $\sum_i a_i c_i \sim k$ in the last loop l_k the effect of the loop is 0. As for other cases, of the effect being negative or positive, the last loop can not be taken infinite number of times.

Note that since the guards are conjunction of constraints of the form $\sum_i a_i c_i \sim k$ and state symbols, after **replacing** the constraints with the corresponding equations as above, we might get more than one equation for one guard in the system.

Consider a counter constraint of the form $\sum_i a_i c_i \sim k$ occuring in the guard $guard(l_i(j))$ in the path schema L. Also, let the loop l_i is taken n times, and the vector of counter values at the transition $l_i(j)$ at corresponding iterations are $(\overrightarrow{x_1}, \overrightarrow{x_2}, \cdots, \overrightarrow{x_n})$.

Claim 2. *For any $\overrightarrow{x} \in \mathbb{N}^n$ and $\Delta \in \mathbb{Z}^n$ for $N \geq 1$ and $\sim \in \{=, <, >, \leq, \geq, \neq\}$, if $\sum_i a_i \overrightarrow{x}[i] \sim k$ and $\sum_i a_i(\overrightarrow{x} + N.\Delta)[i] \sim k$ then $\forall j \in [1, N-1]$, $\sum_i a_i(\overrightarrow{x} + j.\Delta)[i] = \sum_i a_i \overrightarrow{x}[i] + j.\sum_i a_i \Delta[i] \sim k$.*

The proof can be found in the technical appendix.

The combined effect of updates of edges of each loop segment can be summed up as $\Delta \in \mathbb{Z}^n$. Thus, applying the above claim to the case of loop segments, we get that if a term is satisfied at the first and last iteration of a loop, then it is also satisfied in the intermediate iterations of the loop.

For simplicity we replace the variables X_{y_i} with 1, as equations of Type 1 ensures this. Thus, we get, $\mathcal{E}' = \mathcal{E}[1/y_i]$. Clearly, $\mathcal{E}'$ contains only the variables for loops and thus contains only k variables for L . Note that this does not change the solution of the equations, but reduces the number of variables. Thus, if $(m_0, m_1, m_2 \cdots m_{k-1})$ is a solution to $\mathcal{E}'$ then $(1, m_0, 1, m_1, 1, m_2, \cdots, 1, m_{k-1})$ is a solution to $\mathcal{E}$.

Now, we can now prove the following property of the constructed system of equations $\mathcal{E}'$:

Claim 3. *For a given path schema L , for all $\overrightarrow{M} \in \mathbb{N}^k$ $\overrightarrow{M}$ is a solution to $\mathcal{E}'$ iff $\exists \rho$ such that $\delta(\rho) \in fireable(L)$ and $\overrightarrow{M}(\rho) = \overrightarrow{M} \in realizable(L)$*

The proof of the claim can be found in the technical appendix.

5.2 A NP decision procedure for MC(LTL(U,X),FCS)

Here we will prove the following lemma by giving a non-deterministic polynomial-time algorithm in the size of the structure and the formula, which decides MC(LTL(U,X),FCS). We will formulate a system of equations for such runs and find a small solution for such system using the following theorem:

Theorem 5. *[BT76] Let $A \in [-M, M]^{U \times V}$ and $\overrightarrow{b} \in [-M, M]^U$ where $U, V, M \in \mathbb{N}$. If there is $\overrightarrow{x} \in \mathbb{N}^V$ such that $A\overrightarrow{x} \geq \overrightarrow{b}$, then there is $\overrightarrow{y} \in [0, (max\{V, M\})^{CU}]^V$ such that $A\overrightarrow{y} \geq \overrightarrow{b}$, where C is some constant.*

Lemma 4. MC(LTL(U,X),FCS) *is in* NP.

Proof. Let $\varphi \in LTL(U, X)$ and $\mathcal{C}$ be a flat counter system in FCS.

Algorithm

1. Guess a path schema in $\mathcal{C}$, $L = \langle y_0, l_0, y_1, l_1 \cdots, y_k, l_k \rangle$.
2. Guess a vector of integers $\overrightarrow{M} \in [1, 2.td(\phi) + 2]^k$ and we write, $\overrightarrow{M} = (m_0, \cdots, m_{k-1})$.
3. Check if $y_0 l_0^{m_0} y_1 l_1^{m_1} \cdots l_{k-1}^{m_{k-1}} y_k l_k^{\omega} \vDash \varphi$. If not, then abort.
4. Construct the set of formulas $\mathcal{E}$ from L as describe in the Section 5.1.
5. Construct $\mathcal{E}_{\overrightarrow{M}}$ as:

$$\mathcal{E}_{\overrightarrow{M}} = \bigcup_{0 \leq i < k} \begin{cases} X_{l_i} \geq 2.td(\varphi) + 2 & \text{if } m_i = 2.td(\varphi) + 2 \\ X_{l_i} = m_i & \text{otherwise} \end{cases}$$

6. Guess a small polynomial size solution for the system of equations $\mathcal{E} \cup \mathcal{E}_{\overrightarrow{M}}$.
7. Check if the guess satisfies the equations.

Correctness The correctness proof of the algorithm can be found in the technical appendix.

Complexity To prove that the algorithm presented above is in NP we need to show that both the size of all guess done and the running time of the algorithm is a polynome in the size of the system and the size of the formula.

Size of guess: As seen previously, the size of a path schema guessed in Step 1 is bounded by $|\delta|$ and hence by $|\mathcal{C}|$. Also, the size of the vector of integers guessed in Step 2 is at most $k\lceil log_2 2.td(\varphi)+2\rceil$ which is again bounded by both $O(|\varphi|.|\mathcal{C}|)$. Again, as shown in the correctness proof, the guess of solution of system of equation in Step 6 is at most $2k.max(|Q|, max(|\varphi|, |\delta|2^{2|\delta|}))^{C((2n+5)|\delta|)}$. But the encoding of such a solution in binary is bounded by $O(n.|\delta|.(log_2(|Q| + |\varphi| + |\delta|2^{2.|\delta|}))$. Hence the size of the overall guess is a polynome in the size of formula and the size of the system.

Running time: As the guesses are polynomial in size, guessing them in Step 1, 2 and 6 takes polynomial time. As in the proof of Lemma 2, the checking of the satisfaction of φ can be performed in $O(|\mathcal{C}|.|\varphi|)$. As shown previously, the size of the formula $\mathcal{E} \cup \mathcal{E}_{\overrightarrow{M}}$ is $|U| \times (|V| + 1) \times |M|$ which is

$(2k+1).C((2n+5)|\delta|).max(|\varphi|, |\delta|2^{2|\delta|})$. But again this is of polynomial size. Moreover, constructing the equation, requires one pass over the path schema, and hence takes polynomial time. Since the guessed solution of the equation and the equations themselve are of polynomial size, checking the equation, requires addition and multiplication of polynomial number of bits and hence polynomial time. Thus, the whole procedure completes in polynomial amount of time.

Thus, the lemma follows. □

Lemma 5. MC(LTL(U,X),FCS) *is* NP*-hard.*

Proof. The proof follows directly from the proof of Lemma 3, as flat Kripke structures are a specific kind of flat counter systems. □

6 Model Checking LTL with Counter Constraints over Flat Counter Systems

We now consider flat counter systems as before, but now we would like to check $LTL_C(U, X)$ formulas. These are formulas whose atomic propositions are allowed to be linear sum of counter values compared with a fixed constant (e.g. $\sum_i a_i c_i \sim k$). We use an NP procedure that calls the NP procedure developed in the previous section to solve MC($LTL(U, X)$,FCS).

Given a flat counter system $\mathcal{C} = \langle Q, \delta, \Sigma, C, q_{init}\rangle$ with n counters, and a formula $\phi \in LTL_C(U, X)$, we first guess a path schema $L = \langle y_0, l_0, y_1, l_1 \cdots, y_k, l_k\rangle$ in $\mathcal{C}$. Now we would enlarge/modify the path schema, by making copies of the loops l_i in the path schema such that each copy differs in the validity of some of the constraints appearing in ϕ. Thereafter, we construct a new flat counter system $\mathcal{D}$ from the new path schema and transform ϕ to $\phi' \in LTL(U, X)$ such that solving the model checking problem of ϕ' over $\mathcal{D}$ yields solution to the original problem.

6.1 Enlarging path schema by copying loops

By our assumption, we have all the guards and the constraints in the formula concerning the counters of the form $\sum_i a_i c_i \sim k$. We call a *term*, the expression of the form $\sum_i a_i c_i$ occuring on the left side of a constraint. First, we order all such terms arbitrarily and form an arbitrary ordering of terms from the counter system and the formula $\overrightarrow{T} = (\mathtt{t_1}, \mathtt{t_2}, \cdots, \mathtt{t_{m'}})$. Now, we take all such k appearing in the right hand side of each constraint as $K = (k_1, k_2, \cdots, k_m)$ where for all $1 \leq i < m, k_i < k_{i+1}$. Note that, both m and m' are bounded by the size of input (i.e. $|\phi| + |\mathcal{C}|$). Now we can define $(2.m + 1)$ intervals using the elements of K as $I = \{[-\infty, k_1), [k_1, k_1], (k_1, k_2), [k_2, k_2], \cdots, [k_m, k_m], (k_m, \infty)\}$ with a linear ordering of intervals. The loops are copied to make a distinction in the copies of the loop where different constraints are satisfied. With each copy of a loop, we associate properties about terms related to the intervals in I. For making copies of the loops we make the following guesses:

1. The number of times each loop should be copied $\overrightarrow{N} = (n_0, \cdots, n_k)$ (we copy the loop l_i, n_i times and thus replace l_i in L with $\langle l_{(i,1)}, l_i, l_{(i,2)}, \cdots, l_i, l_{(i,n_i)}\rangle$. Intuitively, $l_{(i,j)}$ is the j^{th} copy of i^{th} loop and the k^{th} nodes and transitions of the loop $l_{(i,j)}$ is denoted as $n^k_{(i,j)}$ and $t^k_{(i,j)}$ respectively.

2. A collection of functions $v^k_{(i,j)} : \overrightarrow{T} \to I$ for each $t^k_{(i,j)}$, the k^{th} transition of j^{th} copy of loop l_i, which assigns an interval for each of the terms in $\overrightarrow{T}$, with the following conditions:

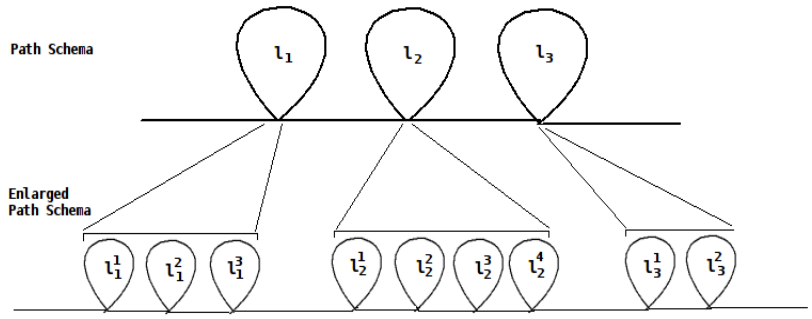


Figure 2: Enlarging path schema by copying loops.

(a) For all $1 \leq h \leq m$, the term $\mathtt{t_h}(\overrightarrow{x}[0], \cdots, \overrightarrow{x}[n-1]) \in v^k_{(i,j)}(\mathtt{t_h})$ for all configurations $(q, \overrightarrow{x})$ in a run $\rho = \cdots (q, \overrightarrow{x})(q', \overrightarrow{x}') \cdots$ where $t^k_{(i,j)}$ is the transition beween $(q, \overrightarrow{x}), (q', \overrightarrow{x}')$. Here, $\mathtt{t_h}(\overrightarrow{x}[0], \cdots, \overrightarrow{x}[n-1])$ denotes the value of $\mathtt{t_h}$ after replacing c_i with the value x_i for all $0 \leq i < n$

(b) For all $0 \leq i \leq k$ and $1 \leq j_1, j_2 \leq m_1$, if $j_1 \neq j_2$ then there exists k and h such that $v^k_{(i,j_1)}(\mathtt{t_h}) \neq v^k_{(i,j_2)}(\mathtt{t_h})$ (i.e. every copy of a loop is distinct from the other with respect to the vectors attached to each transition)

(c) For all $t^k_{i,j}$, we have that $guard(t^k_{i,j})$ is satisfied by some $\overrightarrow{X}$, then $\mathtt{t_h}(\overrightarrow{x}[0], \cdots, \overrightarrow{x}[n-1]) \in v^k_{i,j}(\mathtt{t_h})$, for all terms $\mathtt{t_h} \in \overrightarrow{T}$.

(d) For all $i \in [1, k]$, for all $1 \leq h < m'$, we have either,
- for all $1 \leq j, j' \leq m$, $j < j' \Rightarrow v^k_{i,j}[\mathtt{t_h}] \leq v^k_{i,j'}[\mathtt{t_h}]$, or,
- for all $1 \leq j, j' \leq m$, $j < j' \Rightarrow v^k_{i,j}[\mathtt{t_h}] \geq v^k_{i,j'}[\mathtt{t_h}]$

where for two intervals i_1 and i_2, $i_1 \geq i_2$, iff $max(i_1) \geq max(i_2)$. This conditions ensure that the intervals chosen are monotonous in successive copies of the same loop for the same term.

Thus, we have finally the path schema $L_1 = \langle y_0, l_{(0,1)}, l_0, l_{(0,2)}, \cdots l_{(0,n_1)}, y_1, \cdots, y_t, l_t \rangle$ where each of the transitions $t^k_{(i,j)}$ of each of the loop $l_{(i,j)}$ is associated with a set of guessed intervals $v^k_{(i,j)} = (i_1, i_2, \cdots, i_{m'})$.

Claim 4. *The size of L_1 as constructed above is bounded by a polynome in $|\phi|$ and $|\mathcal{C}|$.*

The proof can be found in the technical appendix.

As L_1 is of polynomial size in the size of the input, the number of intervals we need to guess is also polynomial in the size of the input. Thus, collectively, the guess is of polynomial in the size of the input.

6.2 Construction of counter system and transforming ϕ

Intuitively, next we are going to construct a new counter system $\mathcal{D}$ from L_1 and the guessed intervals. And then transform the given formula $\phi \in LTL_C(U, X)$ to a formula $\phi' \in LTL(U, X)$ with the

constraints removed in such a way that ϕ' is satisfied on $\mathcal{D}$ iff ϕ is satisfied by a path in L_1. In other words, there exists ρ such that $\rho, 0 \vDash \phi$ with $\delta(\rho) \in L$ in $\mathcal{C}$ iff there exists ρ' such that $\rho', 0 \vDash \phi'$ with $\delta(\rho') \in L_1$ in $\mathcal{D}$

For simplification, we rename the loops and paths in L_1 as $\langle y_0, l_0, y_1, l_1 \cdots, y_p, l_p \rangle$ and each path segment σ_i is a sequence of alternate nodes and transitions denoted as below:

$$\begin{aligned} y_i &= (t^1_{y_i}, t^2_{y_i}, \cdots, t^l_{y_i}) \\ l_i &= (t^1_{l_i}, t^2_{l_i}, \cdots, t^l_{l_i}) \end{aligned}$$

The vector of intervals associated to each of the transitions t^j_σ of any path segment σ is denoted by v^j_σ

Transformation of ϕ We define a function f to assign, a unique propositional variable to each constraint as follows:

$$\begin{aligned} f &: Constraints \rightarrow AP \\ f(t \sim b) &= q_{t \sim b} \end{aligned}$$

where, each of the $q_{t\sim b}$ does not appear in the formula (i.e. $q_{t\sim b} \notin AP(\phi)$).

Now,we get, $\phi' = f(\phi)$, where, the function f is extended in the usual way for formulas and $f(\phi)$ denotes relabelling the counter constraints in ϕ to propositional symbols according to the function f. Clearly, $\phi' \in LTL(U, X)$.

Construction of counter system The construction of the new counter system $\mathcal{D}$ is done by labeling the nodes and putting guards in appropriate places. First, in L_1, by the method of assigning the intervals to the transitions, we are assured that the guards on the transitions are satisfied if the transition appears in a run with the terms in the specified intervals associated with the transition. Thus, we can delete all such guards from L_1 and instead ensure that for a transition t^j_σ, all the terms are within the alloted interval of v^j_σ whenever t^j_σ appears at any point of a run. This can be done by putting new guards at some points of the system. Thus the new counter system $\mathcal{D}$ is defined from the given counter system $\mathcal{C} = \langle Q, \delta, \Sigma, C, q_{init} \rangle$ and the schema L_1, as :

$\mathcal{D} = \langle Q, \delta', \Sigma', C, q_{init} \rangle$ where,

- $\Sigma' \subseteq \cup_{\sigma \in S_{L_1}}(\sigma) \times \{\bigwedge_{1 \le h \le m'} \mathtt{t_h} \in v^j_{l_i}(\mathtt{t_h}) | \sigma \in \mathtt{S_{L_1}}, \mathtt{0} \le \mathtt{j} \le |\sigma|\}$ for each term $\mathtt{t_h}$, where $\mathtt{t_h} \in v^j_{l_i}(\mathtt{t_h})$ is the conjunction two conditions as, $\mathtt{t_h} \ge min(v^j_{l_i}(\mathtt{t_h})) \wedge \mathtt{t_h} \le max(v^j_{l_i}(\mathtt{t_h}))$
- δ' is the set of transitions in L_1 (i.e. $\cup_{\sigma \in S_{L_1}} \sigma$) without the guards. New guards are assigned to the transitions as described below.
  1. We add the following guards on the transition t^1_l for each loop l:
     - for each term $\mathtt{t_h}$ and for all $1 \le j < |l|$ we add the constraint $\mathtt{t_h} + \triangle_{=l_{\le j-1}} \in v^j_l(\mathtt{t_h})$ in conjunction. $\bigwedge_{1 \le h \le m'} \mathtt{t_h} + \triangle_{=l_{\le j-1}} \in v^j_l(\mathtt{t_h})$, where for a term $\mathtt{t_h} \stackrel{def}{=} \sum_i a_i.c_i$, we define $\mathtt{t_h} + \triangle_{=\sigma_{\le j-1}} \stackrel{def}{=} \sum_i (a_i.c_i + \triangle_{=\sigma_{\le j}}(i))$. This ensures that when the loop is visited all the terms are in their respective intervals.
  2. If $\delta(l_{i-1}) = \delta(l_i)$ for $1 \le i \le p$, we add conjunction of the following constraints on $t^1_{y_i}$:

– For all transition $t^j_{l_{i-1}}$ in l_i we add $\bigwedge_{1\le h\le m', v^j_{l_{i-1}}(\mathtt{t_h})\neq \mathtt{v}^{\mathtt{j}}_{\mathtt{l_i}}(\mathtt{t_h})} \mathtt{t_h} - \triangle_{=l_{i-1}} + \triangle_{=(l_{i-1})\le j-1} \notin v^j_{l_{i-1}}(\mathtt{t_h})$, where for a term $\mathtt{t_h} \stackrel{def}{=} \sum_i a_i.c_i$, we say $\mathtt{t_h} - \triangle_{=\sigma} + \triangle_{=\sigma_{\le j-1}} \stackrel{def}{=} \sum_i (a_i.c_i - \triangle_{=\sigma}(i) + \triangle_{=\sigma_{\le j-1}}(i))$. This ensures that for the successive copies of same loop, the loop is changed only when some term falls out of an interval.

Given $v : \overrightarrow{T} \to I$ we can define, $v \vDash t \sim b$. Thus, we can say for each constraint $t \sim b$ of the form $\sum_i a_i c_i \sim b$ appearing in the given formula ϕ, whether, $t \sim b$ is true in the interval $v^j_\sigma(t)$. Also note that by the construction of the intervals, a constraint can only be either true or false over the range of a single interval.

The labelling of the system is done by finding out which of the constraints $t \sim b$s are satisfied by any specific interval $v^j_\sigma[i]$ and then putting the propositional symbol $f(t \sim b)$ as the label of $source(t^j_\sigma) = n^j_\sigma$. That is, for constraints of the form $t \sim b$ if $max(v^j_\sigma[i]) \sim b$ and $min(v^j_\sigma[i]) \sim b$ then we assign the label $q_{t\sim b}$ to n^j_σ. Also, note that, we may have more than one propositional symbols as label for each node. This, in particular is not a problem as we can always re-label each node with exactly one propositional symbol and change the formula such that the old propositional symbol q_i is replaced by a disjunction of all the new propositional symbols that replaced q_i from some node. In other words, if at node $n_1, n_2 \cdots n_l$, q_j was replaced by the new propositional symbols $q'_1, q'_2, \cdots q'_l$, then we change the formula ϕ by replacing every occurrance of q_j by $\vee_{i\in[1,l]}(q_i)$. Note that this introduces a blow-up in the size of the formula which is at most polynomial in $|\phi| + |\mathcal{C}|$ as the number of propositional symbols introduced in the formula are bounded by the number of constraints in the formula and the number of such clause is bounded by $|\mathcal{C}|$

Let us denote the new counter system obtained as $\mathcal{D}$ and the transformed formula as ϕ'. Note that we do not put any condition on leaving the last copy of any loop, because the last copy of the loop might be taken any number of times less than the specified number. In spite of this condition, we can still uniquely determine the labels of the nodes of the path segment after the last copy (y_i), as it can be reached in at most two intervals (either in the interval where the last copy was preempted, or the interval after the ones in the labels of the last copy of the loop. Thus, we can consider $\mathcal{D}$ as a counter system on which we want to check the simple LTL formula ϕ'.

6.3 NP decision procedure

Lemma 6. MC($LTL_C(U, X)$,FCS) *is in* NP.

Given a flat counter system $\mathcal{C}$ and a formula $\phi \in LTL_C(U, X)$

1. Guess a path schema $L = \langle y_0, l_0, y_1, l_1 \cdots, y_p, l_p\rangle$
2. Create an enlarged path schema L_1 by making copies of the loops in L with guesses as described in Section 6.1.
3. Transform L_1 into a flat counter system $\mathcal{D}$ and ϕ into $\phi' \in LTL(U, X)$ as described in Section 6.2.
4. Apply the NP decision procedure for MC($LTL(U, X)$,FCS) on $\mathcal{D}$ and ϕ'

Proof. We will show the correctness of the algorithm and the proof that it is in NP.

Correctness The proof of the correctness of the algorithm can be found in the technical appendix.

Complexity The guess of path schema in Step 1, is bounded by $|\mathcal{C}|$. For Step 2, as proved in Claim 4, L_1 is also bounded by a polynomial in $|\mathcal{C}| + |\phi|$. Hence the guesses of interval and number of copy is also bounded by a polynomial in $|\mathcal{C}| + |\phi|$. The transformations in Step 3 requires one pass through L_1 and ϕ, and hence is also bounded by a polynomial in $|\mathcal{C}| + |\phi|$. Also, as shown in the proof of Lemma 4, the NP procedure for MC($LTL(U, X)$, FCS) completes in polynomial time. Thus, the algorithm completes in time bounded by a polynomial in $|\mathcal{C}| + |\phi|$ with guesses of size which is less than $O(|\mathcal{C}|.|\phi|)$. □

NP-hardness of the problem follows directly from the proof of Lemma 3.

Combining the above lemma and the above observation, we can say that,

Theorem 6. *MC($LTL_C(U, X)$,FCS) is NP-complete.*

7 Conclusion

Here, we investigate the model checking problem of various fragments of LTL with counter constraint over various classes of counter systems and establish their optimal complexity bounds. We show that MC($LTL(U, X)$, PS), MC($LTL(U, X)$, FKS), MC($LTL(U, X)$, FCS) and MC($LTL_C(U, X)$, FCS) are all NP-complete. Whereas restricting the model further by fixing the number of loops allowed in a path schema, we obtain that MC($LTL(U, X)$, PS_n) is in P for any fixed $n \geq 1$. The result presents a major improvement over the known 4EXP-time upper bound from [DFGvD10]. The techniques used in the proof, though are well known, but never been combined as is done in the proof.

There are several related problems which are not addressed here, like: extension of the logic with past operators, model-checking for CTL*, model checking over flat counter systems with no restriction over the guards, other extensions of counter systems like affine counter system with finite monoid property [FL02] etc. Though as stated earlier, all these problems are known to be decidable but the upper bounds known for these problems are very crude. Hence it is interesting to investigate the problems to look for tighter upper bounds. Finally, the restrictions on the counter systems may as well be increased to find out when the model checking problem falls below NP.

References

[BFLS06] S. Bardin, A. Finkel, E. Lozes, and A. Sangnier. From pointer systems to counter systems using shape analysis. *Proceedings of the 5th International Workshop on Automated Verification of Infinite-State Systems (AVIS'06)*, 2006.

[BMS+06] M. Bojańczyk, A. Muscholl, Th. Schwentick, L. Segoufin, and C. David. Two-variable logic on words with data. In *LICS'06*, pages 7–16. IEEE, 2006.

[BT76] I. Borosh and L. Treybig. Bounds on positive integral solutions of linear diophantine equations. *American Mathematical Society*, 55:299–304, 1976.

[CGP00] E. Clarke, O. Grumberg, and D. Peled. *Model checking*. The MIT Press Books, 2000.

[CJ98] H. Comon and Y. Jurski. Multiple counters automata, safety analysis and Presburger arithmetic. In *CAV'98*, volume 1427 of *Lecture Notes in Computer Science*, pages 268–279. Springer, 1998.

[Cor02] V. Cortier. About the decision of reachability for register machines. *Theoretical Informatics and Applications*, 36(4):341–358, 2002.

[DFGvD10] S. Demri, A. Finkel, V. Goranko, and G. van Drimmelen. Model-checking CTL* over flat Presburger counter systems. *Journal of Non-Classical Logics*, 20(4):313–344, 2010.

[DL06] S. Demri and R. Lazić. LTL with the freeze quantifier and register automata. In *LICS'06*, pages 17–26. IEEE, 2006.

[EFM99] J. Esparza, A. Finkel, and R. Mayr. On the verification of broadcast protocols. In *LICS'99*, pages 352–359, 1999.

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

[GI81] E. Gurari and O. Ibarra. The complexity of decision problems for finite-turn multicounter machines. In *ICALP'81*, volume 115 of *Lecture Notes in Computer Science*, pages 495–505. Springer, 1981.

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

[KF11] L. Kuhtz and B. Finkbeiner. Weak Kripke structures and LTL. In *CONCUR'11*, Lecture Notes in Computer Science. Springer, 2011. To appear.

[KS05] A. Kučera and J. Strejček. The stuttering principle revisited. *Acta Informatica*, 41(7-8):415–434, 2005.

[Kuh10] L. Kuhtz. *Model Checking Finite Paths and Trees*. PhD thesis, Universitat des Saarlandes, 2010.

[LS05] J. Leroux and G. Sutre. Flat counter systems are everywhere! In *ATVA'05*, volume 3707 of *Lecture Notes in Computer Science*, pages 489–503. Springer, 2005.

[Min67] M. Minsky. *Computation: Finite and Infinite Machines*. Prentice Hall, Englewood Cliffs, NJ, 1967.

[Rac78] C. Rackoff. The covering and boundedness problems for vector addition systems. *Theoretical Computer Science*, 6(2):223–231, 1978.

[Sch03] Ph. Schnoebelen. The complexity of temporal logic model checking. In *AiML'02*, volume 4 of *Advances in Modal Logic*, pages 393–436. King's College, 2003.

Appendices

A Proof of Claim 1

Before proving the Claim, we would like to prove the following lemma:

Lemma 7. *Let $m, n \in \mathbb{N} \cup \{0\}$ and ρ a run such that $w = \delta(\rho) \in \mathcal{L}_L$ where the loop u is (m, n)-redundant. Then, in w,*

1. *u is also (m', n')-redundant for all $m' \in [0, m]$ and $n' \in [0, n]$.*
2. *$u[1]$ is $(m, n-1)$-redundant.*
3. *$u[k]$ is $(m-1, n)$-redundant for all $k \in [0, |u| - 1]$.*

Proof. 1. follows immediately, as the conditions on m' and n' implies that $iter(w, u) \geq (m+1).|u| + n + 1 \geq (m'+1).|u| + n' + 1$. Hence u is (m', n')-redundant.

2. Since u is (m, n)-redundant, its iterations cover at least $(m+1).|u| + n + 1$ letters and thus by the periodicity of the iterated portion of the path, we have that $u[1]$'s iterations covers at least $(m+1).|u| + n$ letters leaving the first letter of the first iteration of u. Note that, the number of *full* iterations of $u[1]$ may be one less, but the last few letters required to cover the specified number of letters after the full iterations are in fact prefix of $u[1]$. So, the last iteration may be incomplete.

3. Similarly as above, if u is (m, n)-redundant, its iterations cover at least $(m+1).|u|+n+1$ letters. Then $u[k]$'s iterations, for $0 \leq k < |u|$ covers at least $(m+1).|u| + n + 1 - k$ letters. This, for the given bound of k implies that $u[k]$'s iteration cover at least $m.|u| + n + 1$ letters and thus, is $(m-1, n)$-redundant. Similar to the previous case, note that the last iteration of $u[k]$ may be incomplete.

□

Proof of Claim 1 First note that though α and β considered here are runs, they can also be considered as control paths as there are no counters in the structure. We will prove the claim using induction on both m and n simultaneously. We consider any formula $\varphi \in LTL(U^m, X^n)$ and two paths $\alpha \rightarrow_{m,n} \beta$ where u is the loop that is (m, n)-redundant and the first iteration of u in α is deleted in β. We show that $\alpha \models \varphi$ iff $\beta \models \varphi$.

Base case. $m = 0$ and $n = 0$. Since u is (m, n)-redundant, it is iterated more than or equal to 2 times in $\delta(\alpha)$ and $\delta(\beta)$ contains u once, with the first occurence deleted. Now, since φ doesn't contain any of the operators U and X, it only contains boolean combination of the state symbols. And thus the satisfiablility of φ depends only on the first node of α and β. Except for the case where u starts from the first node of α, in all other cases the first node of α and β are the same. Hence, we need to consider only this case. But, since u is iterated twice consecutively and only one copy of u was removed. We get that the second copy of u also starts with the same node and thus β starts with the same node as α. Hence it satisfies the induction hypothesis.

Induction Step. Let $m, n \in \mathbb{N} \cup \{0\}$, and let us assume that the claim holds for all m', n' such that either $m' < m$ and $n' \leq n$, or $m' \leq m$ and $n' < n$. We have four possibilities:

1. $\varphi \in LTL(U^{m'}, X^{n'})$ for some m', n' such that either $m' < m$ and $n' \leq n$, or $m' \leq m$ and $n' < n$. By Lemma 7(1) we have u is also (m', n')-redundant, and thus $\alpha \rightarrow_{m',n'} \beta$. By induction hypothesis it follows that $\alpha \models \varphi$ iff $\beta \models \varphi$.

2. $\varphi = X\psi$. We need to prove that $\alpha_1 \models \psi$ iff $\beta_1 \models \psi$. We know that $\psi \in LTL(U^m, X^{n-1})$. So, we just need to show that $\alpha_1 \rightarrow_{m,n-1} \beta_1$, then by induction hypothesis it follows that $\alpha_1 \models \psi$ iff $\beta_1 \models \psi$.

   If u did not start from the first node of α then, like the base case, we have that u is $(m, n-1)$-redundant (by Lemma 7(1)) and thus $\alpha_1 \rightarrow_{m,n-1} \beta_1$. For the case where u start from the first node of α, we get β_1 from α_1 by deleting the loop $u[1]$ as $u[1]$ starts from the first node of α_1. We know from Lemma 7(2) that $u[1]$ is $(m, n-1)$-redundant. Thus, $\alpha_1 \rightarrow_{m,n-1} \beta_1$ and as previous, the result follows by induction.

3. $\varphi = \psi U \eta$. Clearly, $\psi, \eta \in LTL(U^{m-1}, X^n)$. $\alpha \models \psi U \eta$ implies that at some position j, α_j satisfies η and for all $0 \leq i < j$, α_i satisfies ψ.

   We decompose $\delta(\alpha)$ into components as : v - occuring before the first copy of u, y - the iterations of u, w - rest of the path (possibly of infinite length), with $v \in \delta^*$ and $u, y, w \in \delta^+$. Thus, we have, $\delta(\alpha) = vuyw$ and $\delta(\beta) = vyw$. We will first prove that $\beta \models \varphi$ assuming $\alpha \models \varphi$. We deal with three cases, depending on the position of j in α:

   (a) **Before** u : In the case where u starts after the position j in α, we note that for all $0 \leq i \leq j$ we have that $\alpha_i \rightarrow_{m-1,n} \beta_i$, since by Lemma 7(1), u is also $(m-1, n)$-redundant. Thus, η is also satisfied at j in β and ψ is satisfied at all positions before j by the induction hypothesis. Hence, $\beta \models \varphi$.

   (b) **After** u : In this case j occurs after the start of yw in α. Let us consider that the position of j within yw is k. Again, we take the corresponding position k in β (i.e. position $(j - |u|)$ in β). We have, by the same reasoning as previous, that all positions within v in β satisfies ψ. And also, we note that the remaining positions in β occuring before the position k in yw occurs within yw as yw starts just after v in β. Again considering any position i within yw in α, we have the suffix of both the paths starting at i is same. And since we are concerned with only future modalites, they both satisfy the same formulas. Thus, in particular, η is satisfied at k in yw and ψ in all the positions before k within yw. Hence, $\beta \models \varphi$.

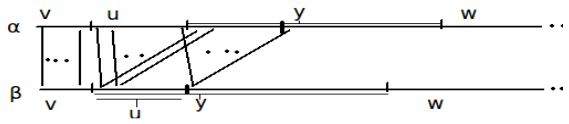
   (c) **Within** u : Since u is deleted in β, we need to find another position k in β, such that $\alpha_j \rightarrow_{m-1,n} \beta_k$ (hence if $\alpha_j \models \eta$ then $\beta_k \models \eta$) and for all $0 \leq i < k, \exists l, l < j$ such that $\alpha_l \rightarrow_{m-1,n} \beta_i$(hence if for all positions before j, ψ is satisfied then for all positions i before k $\beta_i \models \psi$). Then it follows from the induction hypothesis that $\beta \models \varphi$.
   We will prove that for $0 \leq i \leq |v| + |u|, i \leq j, \alpha_i \rightarrow_{m-1,n} \beta_i$. It is clear from the first case that for $0 \leq i \leq |v|$ we have $\alpha_i \rightarrow_{m-1,n} \beta_i$. Now for $|v| < i \leq |v| + |u|$ and $i \leq j$ we note that, β_i is obtained from α_i by deleting $u[i - |v| - 1]$. By Lemma 7(3), we have, $u[i - |v| - 1]$ is $(m-1, n)$-redundant. Hence, for $0 \leq i \leq |v| + |u|, i \leq j, \alpha_i \rightarrow_{m-1,n} \beta_i$ and if some formula (ψ or η) is satisfied at α_i then it is also satisfied at β_i. Thus, $\beta \models \varphi$

   We will now prove that $\alpha \models \varphi$ assuming $\beta \models \varphi$. We deal with three cases again, depending on the position of j, where η is satisfied in β:

(a) **Within** v : In the case where j occurs within v in β, we note that for all $0 \leq i \leq j$ we have that $\alpha_i \rightarrow_{m-1,n} \beta_i$, since by Lemma 7(1), u is also $(m-1,n)$-redundant. Thus, η is also satisfied at j in α and ψ is satisfied at all positions before j by the induction hypothesis. Hence, $\alpha \models \varphi$.

(b) **Within the first iteration of** u **in** y : As proved in the last case of the previous case analysis, we have, for all $0 \leq i \leq j \leq |v| + |u|, \alpha_i \rightarrow_{m-1,n} \beta_i$ and for this case we have $j \leq |v| + |u|$. Thus, we have that in α, η is satisfied at j and ψ is satisfied in all the positions before it as in β. Hence, $\alpha \models \varphi$.

(c) **After the first iteration of** u **in** y : In this case we note that, by previous arguments, for $0 \leq i \leq |v|$ we have $\alpha_i \rightarrow_{m-1,n} \beta_i$. Thus, in all these positions, ψ is satisfied in β and hence ψ is also satisfied in α. Now, observe that for positions $|v| < i \leq |v| + |u|$, by previous case analysis, $\alpha_i \rightarrow_{m-1,n} \beta_i$. Also, by case b. of the previous analysis (reasoning about yw) we have for positions $|v| < i \leq |v| + |u|$, $\alpha_{i+|u|} \rightarrow_{m-1,n} \beta_i$. In this case we have, $j > |v| + |u|$ and thus, for all positions $|v| < i \leq |v|+|u|$ β_i satisfies ψ. Hence, by combining the previous two equivalences we get for positions $|v| < l \leq |v| + |u| + |u|$ α_l satisfies ψ. Also, we know that, for positions $i > |v|+|u|+|u|$ $\alpha_i = \beta_{i-|u|}$ and hence they satisfy same formulas. Thus, we have that $\alpha_{j+|u|}$ satisfies δ and all the prior positions in α satisfies ψ. Hence, $\alpha \models \varphi$.



4. $\varphi = \neg\psi$ or $\varphi = \psi \wedge \delta$. This follows directly from the induction on structure of φ, assuming the induction hypothesis holds for ψ and δ. Thus, $\neg\psi$ is satisfied by α iff it is satisfied by β. Also, $\psi \wedge \delta$ is satisfied by α iff it is satisfied by β, as either of the both is satisfied in one of the path iff it is satisfied in the other path.

B Proof of Lemma 2

For the proof, given a formula φ and a path schema, $L = \langle y_0, l_0, y_1, l_1 \cdots, y_k, l_k \rangle$ in the structure, we guess a vector of integers $\overrightarrow{M} = (m_0, m_1, \cdots, m_{k-1})$, we guess $\overrightarrow{M} \in [1, 2.td(\varphi) + 2]^k$ signifying the number of times the loop l_i should be taken (loop l_k will be taken an infinite number of times). From Theorem 1, it is clear that the each of the m_i 's and in turn $\overrightarrow{M}$ can be bounded by a polynom in the size of the formula φ and the size of the path schema. So, the guess is polynomial in the size of the input. Now, we construct an *ultimately periodic path* P, from the path schema by unfolding each cycle the specified number of times with the exception of the last cycle. An *ultimately periodic path*, uv^ω is defined to be a path with two path segments, where u is a path segment and v is a loop segment which is taken infinite number of times. Clearly, the path P, a sequence of states, is of infinite size, but its encoding can be of size $O(|L|.|\varphi|)$. Now, we transform the given $LTL(U, X)$ formula φ into a CTL formula ψ by introducing $\exists$ quantifiers before every temporal operator. Note that it does not change the models accepted by the formula as we are checking the formula over an infinite *path*, where

CTL and LTL coincide. Also, note that the size of ψ is atmost $2.|\varphi|$. Now, we employ the bilinear algorithm of CTL model checking, see e.g. [Sch03], to check whether the formula ψ is satisfied by the path P in time $O(|P|.|\psi|)$. Thus, we can check whether the given formula φ is satisfied by the given path schema L , in time $O(|L|.|\varphi|^2)$ with a guess of size $O(k.log_2\lceil 2.td(\varphi)+2\rceil)$.

C Proof of Lemma 3

We will prove this by a reduction of the satisfiability problem of propositional logic to the given problem. The satisfiability problem of propositional logic states that, a formula is satisfiable with respect to a class of interpretations if it is possible to find an interpretation that makes the formula true. Given a propositional logic formula f over a finite set of variables $AP = \{b_1, \cdots, b_n\}$. We replace each of the boolean variable in f by a LTL formula which says that, a variable in the propositional formula is set to true, if the corresponding loop is taken once, and false otherwise. We get the LTL formula from f by replacing the occurence of each variable b_i by the LTL formula

$$\bigvee_{i+1 \leq j \leq 3i-1} X^j p_i, \text{ where } p_i \in AP. \tag{1}$$

In effect, we would like to have that a valid control path satisfying the formula over the path schema in Figure 3 gives a satisfying assignment to f. Consider a truth assignment to the variables of f. If b_i is assigned true, then p_i must occur at least once in the control path in the interval $i+1$ to $3i-1$, which amounts to taking the loop of p_i at least once within that interval. Note that it is possible to take the loop in the interval even if we take all the loops for p_j, $j < i$ once or zero times. Otherwise if b_i is assigned false, then we can skip the corresponding loop and make the corresponding LTL formula false. Thus there exists a run in Figure 3 to satisfy the transformed formula, if there exists an assignment to satisfy the propositional formula.

On the other hand, if there exists a control path satisfying the LTL formula, we can assign truth values to the boolean variables by looking at the validity of the corresponding formula on the control path. Note that taking any loop more than once, in a control path amounts to falsifying one of the replaced LTL formula by restricting the occurrance of some p_i outside its interval.

We now give a variant proof of the NP-hardness. This proof ensures that each loop is taken only once and also gives a formula of fixed temporal depth. Instead of Equation 1, we now have the following equation:

$$\bigwedge_i (G(p_i \Rightarrow XG\neg p_i)) \wedge f[Fp_i/b_i] \tag{2}$$

where, $f[Fp_i/b_i]$ represents the transformation of f from a propositional logic formula to a LTL formula by replacing each variable b_i with Fp_i. The proof is same as previous except that the first part of the formula ensures that each loop is taken at most once.

D Proof of Theorem 3

Let us assume the constant restricting the number of loops is n and the formula to be verified is φ. Now, from Theorem 1, we know that each loop can occur at most $2.td(\varphi)+2$. If it occurs more than

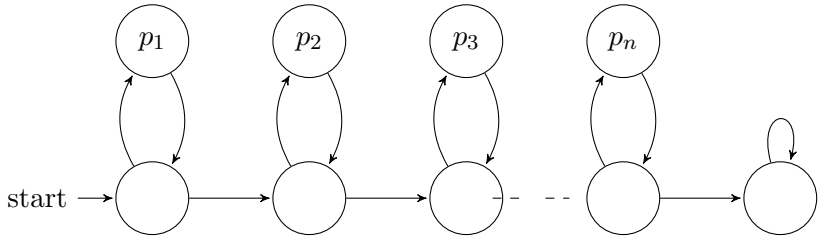


Figure 3: NP-hardness of path schema

that, then they are redundant and can be removed. So, to check whether φ is satisfied or not, we have to check all the ultimately periodic paths obtained by taking each of the cycles from 0 to $2.td(\varphi)+2$ times. So, at most we get $(2.td(\varphi)+2)^n$ paths. Since as previously proved each path is polynomial in the size of the input and model checking can also be performed in polynomial time. The number of paths is exponential in n, but since n is fixed for an instance of the problem, we have that the number of paths $(2.td(\varphi)+2)^n$ is polynomial in the size of the input and hence, the overall procedure completes in P.

E Proof of Claim 3

Consider an infinite run $\rho = (q_0, \overrightarrow{x_0}), \cdots, (q_k, \overrightarrow{x_k}), \cdots$ such that $\delta(\rho) \in \mathcal{L}_L = y_0 l_0^+ y_1 l_1^+ \cdots y_k l_k^\omega$. If $\overrightarrow{M}(\rho) = \overrightarrow{M} = (m_0, m_1, \cdots, m_{k-1})$ with each $m_i \geq 1$ then we need to prove that $\overrightarrow{M}$ is a solution to $\mathcal{E}'$

1. Since $\delta(\rho) \in \mathcal{L}_L$, each simple segment y_i is taken only once and hence $\overrightarrow{M}$ satisfies all equations of Type 1 in $\mathcal{E}'$.

2. Again, since ρ is a valid run, every guard of the transition are satisfied in ρ. Also, since $\delta(\rho) \in \mathcal{L}_L$, every segment is taken at least once. Hence $\overrightarrow{M}$ satisfies all equations of Type 2 in $\mathcal{E}'$.

3. Since ρ is a valid run, every guard of the transition are satisfied in ρ. Also, every segment appears at least once. Thus, by the validity of ρ , we have that all the guards on the transitions are satisfied at least once. Hence $\overrightarrow{M}$ satisfies all equations of Type 3 in $\mathcal{E}'$.

4. Again, the guards on the edges for the last iteration also appear in ρ and are satisfied in the last iteration also. Thus, $\overrightarrow{M}$ satisfies all equations of Type 4.

5. Since ρ is a valid path, the counter values never reach negative. Hence $\overrightarrow{M}$ satisfies all equations of Type 5 in $\mathcal{E}'$.

6. Again by validiy of ρ , the last loop is taken infinite number of times. So, $\overrightarrow{M}$ satisfies all equations of Type 6.

Since, for every transition, $\overrightarrow{M}$ satisfies all type of formulas in $\mathcal{E}'$, we have that $\overrightarrow{M}$ is a solution to $\mathcal{E}'$.

On the other hand, consider a vector of integers $\overrightarrow{M} = (m_0, m_1, \cdots, m_{k-1})$ such that $\overrightarrow{M}$ is a solution to $\mathcal{E}'$. Now, from the given path schema L and $\overrightarrow{M}$, we can construct $\rho = (q_0, \overrightarrow{x_0}), \cdots, (q_k, \overrightarrow{x_k}, \cdots$ with

$\overrightarrow{x_0}$) a zero vector and all other $\overrightarrow{x_k}$ are defined from the update functions such that $\overrightarrow{M}(\rho) = \overrightarrow{M}$ and $\delta(\rho) \in \mathcal{L}_L$. Note that ρ is not necessarily a run, since it may happen that in ρ, each $x_i \in \mathbb{Z}^n$. We would show that $\delta(\rho) \in fireable(L)$ and hence ρ is indeed a run. Since $\overrightarrow{M}$ is a solution to $\mathcal{E}'$, $\overrightarrow{M}$ satisfies all the formulas in $\mathcal{E}'$:

- Formulas of Type 1 and Type 2, ensures that $\delta(\rho) \in \mathcal{L}_L$
- Formulas of Type 5 ensure that for all $\overrightarrow{x_k} \in \mathbb{N}^n$ and in fact $\rho \in (Q \times \mathbb{N}^n)^\omega$.
- Formulas of Type 3 ensure that counter values at any point in ρ satisfies the guards of the next transition and hence allowing to take the transition for the first copy of any segment in ρ.
- Formulas of Type 4 ensure that the counter values at the last repetition of any segment satisfy the guards on the transitions. As proved in Claim 2, this ensures that in all the repetition of a segment, the counter values are valid with respect to the guards. Hence, ρ contains every loop designated number of times and again all the guards on the transitions are satisfied.
- Formulas of Type 6 ensure that during the infinite number of iterations of last loop, the guards are not violated.

In effect, the equations ensure that for every configuration in ρ, $\overrightarrow{x_k}$ satisfies the guard on the transition $((q_k, \overrightarrow{x_k}), (q_{k+1}, \overrightarrow{x_{k+1}}))$ and $\rho \in (Q \times \mathbb{N}^n)^\omega$. Hence ρ is a run such that $\delta(\rho) \in fireable(L)$.

F Correctness proof of algorithm of Lemma 4

Let us define a function:

$$truncate : \mathbb{N} \to [1, 2.td(\varphi) + 2]$$
$$truncate(i) = \begin{cases} 2.td(\varphi) + 2 & \text{if } i \geq 2.td(\varphi) + 2 \\ i & \text{otherwise} \end{cases}$$

The function truncate is extended to vector of integers as $truncate(\overrightarrow{M})[i] = truncate(\overrightarrow{M}[i])$ for $i \in [1, k]$. Now, let us consider, a run $\rho = (q_0, \overrightarrow{x_0}), \cdots, (q_k, \overrightarrow{x_k}), \cdots$ such that $\rho \vDash \varphi$. From ρ we get L such that $\delta(\rho) \in fireable(L)$ and a vector of integers $\overrightarrow{M}(\rho) = (i_0, \cdots, i_{k-1}) \in realizable(L)$.

1. Since we non-deterministically guess the path schema, let L is the guessed path schema.
2. Since we guess the integers non-deterministically, without loss of generality we may assume that the guessed vector $\overrightarrow{M} = (j_0, j_1 \cdots j_{k-1}) = truncate(i_0, i_1 \cdots i_{k-1})$
3. If $y_0 l_0^{j_0} y_1 l_1^{j_1} \cdots l_{k-1}^{j_{k-1}} y_k l_k^\omega \vDash \varphi$ then $y_0 l_0^{i_0} y_1 l_1^{i_1} \cdots l_{k-1}^{i_{k-1}} y_k l_k^\omega \vDash \varphi$ as proved in Theorem 1.
4. By Claim 3, $(i_0, i_1, \cdots, i_{k-1})$ is a solution to $\mathcal{E}'$.
5. Formula $\mathcal{E}_{\overrightarrow{M}}$ ensures that if $(i_0, i_1, \cdots, i_{k-1})$ is a solution to $\mathcal{E}_{\overrightarrow{M}}$ then $\overrightarrow{M} = truncate(i_0, i_1 \cdots i_{k-1})$

6. Since ρ is a valid configuration path and since $\overrightarrow{M} = truncate(i_0, i_1 \cdots i_{k-1})$, $\rho \vDash \varphi$, by Claim 3, $(i_0, i_1 \cdots i_{k-1})$ is a valid assignment to the variables $X_{l_0}, X_{l_1} \cdots X_{l_{k-1}}$ In this case, the existence of a small polynomial size solution of $\mathcal{E} \cup \mathcal{E}_{\overrightarrow{M}}$ is ensured by looking at the formulas as system of equations and applying Theorem 5. Here, we take $|\delta|$ to be the size of the encoding of the set of transitions δ. By application of the Theorem 5 we get:

   (a) V is number of variables in the system, which in our case is the number of segments in the path schema(i.e. $2k$). This value is bounded by the number of transitions in the structure. Thus, we have $|V| \leq |\delta|$.

   (b) M is maximum coefficient appearing in the equations. For equations of Type 1 and 2, the maximum coefficient is 1. For equations of Type 3 and 4, the maximum coefficient of any variable is bounded by maximum update by any segment and maximum a_i occuring in any guard. This is bounded by $(|\delta|.2^{|\delta|}).2^{|\delta|}$ as these are encoded in binary in the definition of δ. Again, the maximum constant on right side is bounded by $2^{|\delta|}$. For equations of Type 6, the maximum constant that can appear is bounded by $2^{|\delta|}.2^{|\delta|}$. For formulas in $\mathcal{E}_{\overrightarrow{M}}$, the maximum constant is bounded by $2.td(\phi) + 2$ which is turn is bounded by $|\phi|$. Thus, M in our case is bounded by $max(|\varphi|, |\delta|2^{2|\delta|})$.

   (c) U is the number of equations. There are k equations of Type 1 and 2, which is bounded by $|\delta|$. Also, $|\delta|$ equations each of Type 3 and 4. Moreover, $2n|\delta|$ and $|\delta|$ equations of Type 5 and 6 respectively. And, at most k equations in $\mathcal{E}_{\overrightarrow{M}}$, which again is bounded by $|\delta|$. Thus, we have, $U = (2n + 5)|\delta|$

   From the theorem, we have that if there is a solution to the equations, there is one in the interval $[0, max(|Q|, max(|\varphi|, |\delta|2^{2|\delta|}))^{C((2n+5)|\delta|)}]^{2k}$. Thus, we can guess a solution of $\mathcal{E} \cup \mathcal{E}_{\overrightarrow{M}}$ which is at most exponential in the size of the input, but such a solution requires just polynomial number of bits to represent as we are encoding the solution in binary.

7. Again since we guess the solution to $\mathcal{E} \cup \mathcal{E}_{\overrightarrow{M}}$ non-deterministically, without loss of generality we may assume that the guessed solution $(g_0, h_0, g_1, h_1 \cdots g_{k-1}, h_{k-1})$ such that $g_i = 1, \forall 0 \leq i < k$ and $(j_0, j_1 \cdots j_{k-1}) = truncate(h_0, h_1 \cdots h_{k-1}) = truncate(i_0, i_1 \cdots i_{k-1})$

Since there exists a solution to $\mathcal{E} \cup \mathcal{E}_{\overrightarrow{M}}$, the algorithm completes successfully and returns witness.

For the other direction, consider a witness output by the algorithm. It consists of a path schema L , a vector of integers $\overrightarrow{M} = (m_0, \cdots, m_{k-1})$ and a small solution $\overrightarrow{n} = (n'_0, n_0, n'_1, n_1 \cdots n'_{k-1}, n_{k-1}))$ for $\mathcal{E} \cup \mathcal{E}_{\overrightarrow{M}}$. We would construct a run ρ such that $\rho, 0 \vDash \varphi$.

From the given path schema L and $\overrightarrow{n}$, we can construct $\rho = (q_0, \overrightarrow{x_0}), \cdots, (q_k, \overrightarrow{x_k}), \cdots$ with $\overrightarrow{x_0}$ is a zero vector and all other $\overrightarrow{x_k}$ are defined from the update functions and $\overrightarrow{M}(\rho) = (n_0, n_1 \cdots, n_{k-1})$ and $\delta(\rho) \in \mathcal{L}_L$. Since the $\overrightarrow{n}$ is a solution to $\mathcal{E} \cup \mathcal{E}_{\overrightarrow{M}}$, we know by Claim 3, that ρ is a valid run and $truncate((n_0, n_1, \cdots n_{k-1})) = \overrightarrow{M}$. Also, since $\overrightarrow{M}$ is one of the witness of the algorithm, we have that $y_0 l_0^{\overrightarrow{M}[0]} y_1 l_1^{\overrightarrow{M}[1]} \cdots l_{k-1}^{\overrightarrow{M}[k-1]} y_k l_k^{\omega} \vDash \varphi$. Hence, by Theorem 1, $\rho, 0 \vDash \varphi$.

G Proof of Claim 2

Clearly $f(\overrightarrow{x}) \stackrel{def}{=} \sum_i a_i \overrightarrow{x}[i]$ is a either a monotonically increasing or monotonically decreasing function. Let us denote by max and min as $max(\overrightarrow{x}, \overrightarrow{x} + N.\Delta)$ and $min(\overrightarrow{x}, \overrightarrow{x} + N.\Delta)$ respectively, where for two vectors $\overrightarrow{x}, \overrightarrow{y}$, $max(\overrightarrow{x}, \overrightarrow{y})$ (respectively $min(\overrightarrow{x}, \overrightarrow{y})$) is defined to be x (respectively y) if for all $0 \leq i \leq |\overrightarrow{x}|, \overrightarrow{x}(i) \geq \overrightarrow{y}(i)$. Also, max_f and min_f are defined as $max(f(\overrightarrow{x}), f(\overrightarrow{x} + N.\Delta))$ and $min(f(\overrightarrow{x}), f(\overrightarrow{x} + N.\Delta))$ respectively. Hence, $\forall j \in [1, N-1]$, $f(\overrightarrow{x} + j.\Delta)[i]) \in [min_f, max_f]$ since, $\overrightarrow{x} + j.\Delta \in [min, max]$. Hence the claim holds.

H Proof of Claim 4

First note that, the size of L as mentioned in the text, and the number of loops in L is bounded by a polynomial in $|\phi| + |\mathcal{C}|$. Also, the number of intervals $|I|$ is bounded by $|\phi| + |\mathcal{C}|$. Consider any loop l_i in L . The number of nodes in l_i is also bounded by $|\mathcal{C}|$. The number of intervals m' associated with any node of l_i is also bounded by $|\phi| + |\mathcal{C}|$. Now, each of this intervals can independently change only $|I|$ times because of the monotonicity of updates in a loop as discussed in the proof of Claim 2. Hence, the maximum number of copies of l_i can be $|I|.m'.|l_i|.|L|$ which is bounded by $(|\phi| + |\mathcal{C}|)^2.|\mathcal{C}|^2$ which is the required bound.

I Correctness proof of algorithm of Lemma 6

First we consider a run $\rho = (q_0, \overrightarrow{x_0}), \cdots, (q_k, \overrightarrow{x_k}), \cdots$ in $\mathcal{C}$, such that $\rho, 0 \vDash \phi$. We will show that if $\rho, 0 \vDash \phi$, then there exists an accepting computation of the above algorithm.

1. Since there exists a run $\rho \vDash \phi$, we can get a path schema L in $\mathcal{C}$ such that $\delta(\rho) \in L$ and we also get a vector of integers $\overrightarrow{M} = (m_0, m_1, \cdots, m_k)$, where m_i signifies the number of times l_i is taken in ρ . Since, the path schema is guessed non-deterministically, without loss of generality, we may assume that L is the guessed path schema.

2. Now, for each m_i, we can decompose it into $m_i = m_i^1 + 1 + m_i^2 + 1 + m_i^3 + \cdots + m_i^k$ such that if $(q_i^j, \overrightarrow{x_i}^j) \xrightarrow{m_i^j.l_i} (q_k^{j+m_i^j.|l_i|}, \overrightarrow{x_k}^{j+m_i^j.|l_i|})$ denotes the part of the run covered by l_i taken m_i^j times then $\forall \mathtt{t_k} \in \overrightarrow{T}, \mathtt{t_k}(\overrightarrow{x_i}^j) \in I[k] \Rightarrow \forall 1 \leq h < m_i^{j.|l_i|} \mathtt{t_k}(\overrightarrow{x_i}^{j+h}) \in I[k]$ and for at least one k, $\mathtt{t_k}(\overrightarrow{x_i}^j) \in I[k] \Rightarrow \mathtt{t_k}(\overrightarrow{x_i}^{j+m_i^j.|l_i|}) \notin I[k]$ where $\mathtt{t_k}(\overrightarrow{x})$ is as defined earlier to be the value of term $\mathtt{t_k}$ where the counter values are substituted from $\overrightarrow{x}$. Again, due to nondeterminism of our guess of the new path schema L_1 we may assume without loss of generality, that each loop l_i is copied exactly the number of times m_i is partitioned. And the intervals are guessed accordingly. Note that, L_1 and the partitions of m_i give us the same run ρ .

3. The construction of the counter system $\mathcal{D}$ ensures that, there exists $\rho\prime$ in $\mathcal{D}$ such that if $\rho[i] = (q_i, \overrightarrow{x_i})$ and $\overrightarrow{x_i} \vDash t \sim b$ then $\rho'[i] = (q_i', \overrightarrow{x})$ and $q_i' = q_{t \sim b}$ where $\rho[i]$ for a run ρ denotes the i^{th} configuration in ρ . Thus, if $\rho, 0 \vDash \phi$ then $\exists \rho', \rho', 0 \vDash f(\phi) = \phi'$.

4. Since, $\exists \rho', \rho', 0 \vDash \phi'$, the NP decision procedure for MC($LTL(U, X)$,FCS) on $\mathcal{D}$ and ϕ' and hence the present procedure will return true.

Now for the other side, we consider that the given procedure returns true.

- Since the algorithm returns true, we know that the NP decision procedure for MC($LTL(U, X)$,FCS) on $\mathcal{D}$ and ϕ' and hence $\exists \rho', \rho', 0 \vDash \phi'$. Again from ρ' we can get a path schema L' and a vector of integers $\overrightarrow{M'}$ such that $\delta(\rho') \in L'$.
- By construction of $\mathcal{D}$ and non-determinism in guessing of L_1 we may assume that, if $\rho'[i] = (q'_i, \overrightarrow{x_i})$ and $q'_i = q_{t \sim b}$ then $\exists \rho$, such that $\delta(\rho) \in L_1$ and $\rho[i] = (q_i, \overrightarrow{x})$ and $\overrightarrow{x_i} \vDash t \sim b$. Hence, $\rho, 0 \vDash \phi$. Again, ρ and L_1 gives us a vector of integers $\overrightarrow{M_1}$ as before.
- We can collaspe the loops l_i and l_j in L_1 if $l_i = l_j$ and add the corresponding integers too. Again, by the non-determinism in guessing the path schema and by the construction of L_1 we get the path schema L . Note that $\delta(\rho) \in L$.
- L is a path schema in $\mathcal{C}$. Hence, $\exists \rho$ in $\mathcal{C}$ such that, $\rho, 0 \vDash \phi$