How Hard is It to Verify Flat Affine Counter Systems with the Finite Monoid Property?

Radu Iosif, Arnaud Sangnier

How hard is it to verify flat affine counter systems with the finite monoid property ?

Radu Iosif[1] and Arnaud Sangnier[2]

[1] Verimag, Univ Grenoble Alpes, CNRS
[2] IRIF, Univ Paris Diderot, CNRS

Abstract. We study several decision problems for counter systems with guards defined by convex polyhedra and updates defined by affine transformations. In general, the reachability problem is undecidable for such systems. Decidability can be achieved by imposing two restrictions: (1) the control structure of the counter system is *flat*, meaning that nested loops are forbidden, and (2) the multiplicative monoid generated by the affine update matrices present in the system is finite. We provide complexity bounds for several decision problems of such systems, by proving that reachability and model checking for Past Linear Temporal Logic stands in the second level of the polynomial hierarchy Σ_2^P, while model checking for First Order Logic is PSPACE-complete.

1 Introduction

Counter systems are finite state automata extended with integer variables, also known as counter automata or counter machines. These are Turing-complete models of computation, often used to describe the behavior of complex real-life systems, such as embedded/control hardware and/or software systems. Because many verification problems, of rather complex systems, can be reduced to decision problems for counter systems, it is important to understand the difficulties faced by potential verification algorithms designed to work with the latter.

Due to their succinctness and expressive power, most decision problems, such as reachability, termination and temporal logic model-checking, are undecidable for counter systems, even when the operations on the counters are restricted to increment, decrement and zero-test [24]. This early negative result motivated the search for subclasses with decidable decision problems. Such classes include *one-counter systems* [14], *vector addition systems with states* [22], *reversal-bounded counter machines* [15] and *flat counter systems* [4,12].

Flat counter systems are defined by a natural syntactic restriction, which requires that no state occurs in more than one simple cycle in the control flow graph of the system. Decidability results on the verification of reachability problems for flat counter systems have been obtained by proving that, under certain restrictions on the logic that defines the transition rules, the set of reachable configurations is semilinear and effectively definable in Presburger arithmetic [4,12,6]. Even though flatness is an important restriction (few counter systems

modeling real-life hardware and software artifacts are actually flat), this class provides the grounds for a useful method that under-approximates the set of behaviors of a non-flat counter system by larger and larger sets of paths described by flat counter systems. This method is currently used by model checking tools, such as FAST [2] and FLATA [19], and has been applied to improve the results of static analysis [13], as well as the convergence of counterexample-driven abstraction refinement algorithms [17]. Moreover, several works define classes of *flattable* counter systems, for which there exist flat unfoldings of the system with identical reachability sets. Such is the case of timed automata [7] and of 2-dimensional vector addition systems with states [21,3]. For these systems, the method of under-approximations by flat unfoldings is guaranteed to terminate.

In general, the flatness restriction is shown to reduce the computational complexity of several decision problems, such as reachability or temporal logic model checking. For instance, in the case of Kripke structures, flatness reduces the complexity of the model-checking of Linear Temporal Logic (LTL) from PSPACE to NP [20]. When considering flat counter systems whose updates are described by translations, the complexity of these problems drops from undecidable to NP-complete [10], while model checking for First Order Logic (FO) is coined to be PSPACE-complete [8]. For branching time temporal logics, flatness yields decidable problems, but with less remarkable complexity bounds [9].

In this work, we focus on the model of affine counter systems, in which each transition is labeled with (i) a guard defined by (a disjunction of) convex polyhedra, i.e. linear systems of inequalities of the form $\boldsymbol{C} \cdot \boldsymbol{x} \leq \boldsymbol{d}$, and (ii) a deterministic update defined by an affine transformations $f(\boldsymbol{x}) = \boldsymbol{A} \cdot \boldsymbol{x} + \boldsymbol{b}$ where $\boldsymbol{A}, \boldsymbol{C} \in \mathbb{Z}^{n \times n}$ are square matrices with integer entries, $\boldsymbol{b}, \boldsymbol{d} \in \mathbb{Z}^n$ are vectors of integer constants and $\boldsymbol{x} = [x_1, \ldots, x_n]$ is a vector of counters. For such systems, the set of reachable configurations is semilinear (thus reachability is decidable), provided that the multiplicative monoid generated by the matrices used in update functions is finite. This condition is also known as the *finite monoid property* [4,12]. Moreover, it has been shown that the model-checking of such systems, for an extended version of the branching time logic CTL^* is decidable, also by reduction to the satisfiability of a Presburger formula, of size exponential in the size of the counter system [11].

In this work, we show that for flat affine counter systems with the finite monoid property, reachability and model checking for Past LTL are Σ_2^P, whereas model checking for FO is PSPACE-complete. Our result generalizes the results for flat counter systems with translations [8,10], since these systems are a strict subclass of flat affine counter systems with the finite monoid property. For instance, a transfer of values between different counters can be done in one step with an affine counter system, whereas a translating counter system would need a cycle to implement such operations. Our proof technique is based on an analysis of the behavior of the sequence of matrix powers in a finite multiplicative monoid, and adapts several techniques for translating counter systems to this more general case.

Due to lack of space, omitted proofs can be found in [18].

2 Counter Systems and their Decision Problems

We denote by $\mathbb{N}$ and $\mathbb{Z}$ the sets of natural and integer numbers, respectively. We write $[\ell, u]$ for the integer interval $\{\ell, \ell+1, \ldots, u\}$, where $\ell \leq u$, and $\mathrm{abs}(n)$ for the absolute value of the integer $n \in \mathbb{Z}$. The cardinality of a finite set S is denoted by $\|S\|$.

We denote by $\mathbb{Z}^{n\times m}$ the set of matrices with n rows and m columns, where $\boldsymbol{A}[i]$ is the i-th column and $\boldsymbol{A}[i][j]$ is the entry on the i-th row and j-th column of $\boldsymbol{A} \in \mathbb{Z}^{n\times m}$, for each $i \in [1, n]$ and $j \in [1, m]$. If $n = m$, we call this number the *dimension* of A, and we denote by $\mathbf{I}_n$ the identity matrix in $\mathbb{Z}^{n\times n}$. For $\boldsymbol{A} \in \mathbb{Z}^{n\times m}$ and $\boldsymbol{B} \in \mathbb{Z}^{m\times p}$, we denote by $\boldsymbol{A} \cdot \boldsymbol{B} \in \mathbb{Z}^{n\times p}$ the matrix product of $\boldsymbol{A}$ and $\boldsymbol{B}$. For a matrix $\boldsymbol{A} \in \mathbb{Z}^{n\times n}$, we define $\boldsymbol{A}^0 = \mathbf{I}_n$ and $\boldsymbol{A}^i = \boldsymbol{A}^{i-1} \cdot \boldsymbol{A}$, for all $i > 0$.

We write $\mathbb{Z}^n$ for $\mathbb{Z}^{n\times 1}$ in the following. Each $\boldsymbol{v} \in \mathbb{Z}^n$ is a column vector, where $\boldsymbol{v}[i]$ is the entry on its i-th row. For a vector $\boldsymbol{x}$ of variables of length n and a matrix $\boldsymbol{A} \in \mathbb{Z}^{m\times n}$, the product $\boldsymbol{A} \cdot \boldsymbol{x}$ is the vector of terms $(\boldsymbol{A} \cdot \boldsymbol{x})[i] = \sum_{j=1}^{n} \boldsymbol{A}[i][j] \cdot \boldsymbol{x}[j]$, for all $i \in [1, m]$. A row vector is denoted by $\boldsymbol{v} = [v_1, \ldots, v_n] \in \mathbb{Z}^{1\times n}$. For a row vector $\boldsymbol{v}$, we denote its transpose by $\boldsymbol{v}^\top$.

For a vector $\boldsymbol{v} \in \mathbb{Z}^n$, we consider the standard infinity $\|\boldsymbol{v}\|_\infty = \max_{i=1}^{n} \mathrm{abs}(\boldsymbol{v}[i])$ norm . Given $\boldsymbol{A} \in \mathbb{Z}^{m\times n}$, consider the induced $\|\boldsymbol{A}\|_\infty = \max_{i=1}^{m} \sum_{j=1}^{n} \mathrm{abs}(\boldsymbol{A}[i][j])$, and the maximum $\|\boldsymbol{A}\|_{\max} = \max_{i=1}^{m} \max_{j=1}^{n} \mathrm{abs}(\boldsymbol{A}[i][j])$ norms. The size of a matrix is $size(\boldsymbol{A}) = \sum_{i=1}^{m} \sum_{j=1}^{n} \log_2(\boldsymbol{A}[i][j]+1)$, with integers encoded in binary.

2.1 Counter systems

Let $\mathsf{X}_n = \{x_1, x_2, \ldots, x_n\}$ be a finite set of integer variables, called *counters*, $\boldsymbol{x}$ be the vector such that $\boldsymbol{x}[i] = x_i$, for all $i \in [1, n]$, and $\mathsf{AP} = \{\mathrm{a}, \mathrm{b}, \mathrm{c}, \ldots\}$ be a countable set of boolean *atomic propositions*. A *guard* is either true, denoted by $\top$, or a disjunction of systems of inequalities, denoted by $\bigvee_{i=1}^{k} \boldsymbol{C}_i \cdot \boldsymbol{x} \leq \boldsymbol{d}_i$ where $\boldsymbol{C}_i \in \mathbb{Z}^{m\times n}$ and $\boldsymbol{d}_i \in \mathbb{Z}^m$ for all $i \in [1, k]$. A guard is said to be without disjunction if it is either true or it consists of a single system of inequalities.

An integer vector $\boldsymbol{v} \in \mathbb{Z}^n$ *satisfies* the guard g, written $\boldsymbol{v} \models \mathsf{g}$, if either (i) $\mathsf{g} := \top$, or (ii) $\mathsf{g} := \bigvee_{i=1}^{k} \boldsymbol{C}_i \cdot \boldsymbol{x} \leq \boldsymbol{d}_i$ and $\boldsymbol{v}$ is a solution of a system $\boldsymbol{C}_i \cdot \boldsymbol{x} \leq \boldsymbol{d}_i$, for some $i \in [1, k]$. The set of guards using X_n is denoted by $\mathsf{CG}(\mathsf{X}_n)$. An *affine function* $f : \mathbb{Z}^n \to \mathbb{Z}^n$ is a pair $(\boldsymbol{A}, \boldsymbol{b}) \in \mathbb{Z}^{n\times n} \times \mathbb{Z}^n$. Given a vector $\boldsymbol{v} \in \mathbb{Z}^n$, the result of the function $f = (\boldsymbol{A}, \boldsymbol{b})$ applied to $\boldsymbol{v}$ is $f(\boldsymbol{v}) = \boldsymbol{A} \cdot \boldsymbol{v} + \boldsymbol{b}$. We denote by Aff_n the set of affine functions over $\mathbb{Z}^n$. An affine function $(\boldsymbol{A}, \boldsymbol{b})$ where $\boldsymbol{A} = \boldsymbol{I}_n$ is called a *translation*.

Definition 1. *[Affine Counter System] For an integer $n \geq 0$, an* affine counter system of dimension n *(shortly a counter system) is a tuple $S = \langle Q, \mathsf{X}_n, \Delta, \Lambda \rangle$, where: (i) Q is a finite set of* control states*, (ii) $\Lambda : Q \to 2^{\mathsf{AP}}$ is a* labeling function*, and (iii) $\Delta \subseteq Q \times \mathsf{CG}(\mathsf{X}_n) \times \mathsf{Aff}_n \times Q$ is a finite set of* transition rules *labeled by guards and affine functions (updates).*

A counter system is said to be *disjunction free* if all its guards are without disjunction. For a transition rule $\delta = \langle q, \mathsf{g}, f, q' \rangle \in \Delta$, we use the notations

$source(\delta) = q$, $guard(\delta) = \mathsf{g}$, $update(\delta) = f$ and $target(\delta) = q'$. A *path* π of S is a non-empty sequence of transition rules $\delta_1 \dots \delta_m$ such that $source(\delta_{i+1}) = target(\delta_i)$ for all $i \in [1, m-1]$. The path π is a *simple cycle* if $\delta_1 \dots \delta_m$ are pairwise distinct and $source(\delta_1) = target(\delta_m)$. In this case, we denote $source(\pi) = target(\pi) = source(\delta_1)$. A counter system S is *flat* if for each control state $q \in Q$ there exists at most one simple cycle π such that $source(\pi) = q$. In such a system any path leaving a simple cycle cannot revisit it.

Example 1. Figure 1 shows a flat counter system whose control states q_0, q_1, q_2, q_3 are labeled by the atomic propositions $\mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{d}$, respectively. From the initial state q_0 with all counters equal to 0, this system begins with incrementing x_1 a certain number of times by a transition δ_0 then, with δ_1, it transfers the value of the counter x_1 to x_3 and resets x_1; the loop labeled by δ_2 increments both x_1 and x_2 until they both reach the value of x_3 and finally the loop labeled by δ_4 is used to decrement x_2 and increment x_1 until the value of x_1 is twice the value of x_3. As a consequence, when the system reaches q_3 the value of x_1 is twice the value of x_3 and the value of x_2 is equal to 0. Hence, any run reaching q_3 visits the state q_1 exactly the same number of times as the state q_2. ■

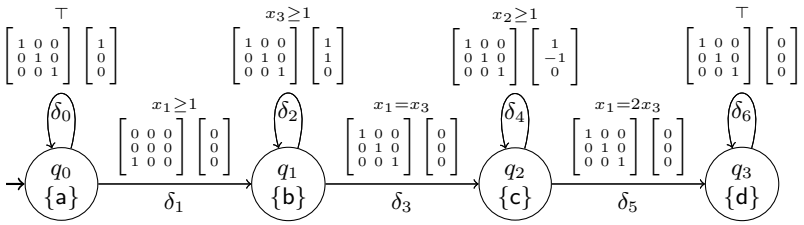

Fig. 1. A flat affine counter system

The size of a counter system S is $size(S) = \sum_{\delta \in \Delta} size(\delta) + \sum_{q \in Q} \|\Lambda(q)\|$, where $size(\delta) = 1 + size(guard(\delta)) + size(update(\delta))$, for a guard $\mathsf{g} := \bigvee_{i=1}^{k} \boldsymbol{C}_i \cdot \boldsymbol{x} \leq \boldsymbol{d}_i$ we have $size(g) = \Sigma_{i=1}^{k} size(\boldsymbol{C}_i) + size(\boldsymbol{d}_i)$, and for an update $f = (\boldsymbol{A}, \boldsymbol{b})$, $size(f) = size(\boldsymbol{A}) + size(\boldsymbol{b})$.

A counter system of dimension $n = 0$ is called a *Kripke structure*. We denote by KS and KS_f the sets of Kripke structures and flat Kripke structures, respectively. A counter system of dimension $n \geq 1$ is *translating* if all updates labeling the transition rules are pairs $(\mathbf{I}_n, \boldsymbol{b})$. Let TS and TS_f denote the sets of translating and flat translating counter systems of any dimension $n \geq 1$.

For a counter system S of dimension $n \geq 1$, we consider $\mathcal{M}_S \subseteq \mathbb{Z}^{n \times n}$ to be the smallest set of matrices, closed under product, which contains $\mathbf{I}_n$ and each matrix $\boldsymbol{A}$ occurring in an update $(\boldsymbol{A}, \boldsymbol{b})$ of a transition rule in S. Clearly, $\mathcal{M}_S$ forms a monoid with the matrix product and identity $\mathbf{I}_n$. We say that S has the

finite monoid property if the set $\mathcal{M}_S$ is finite. Let $\mathsf{AS}_{\mathrm{fm}}$ be the set of flat counter systems with the finite monoid property and $\mathsf{AS}_{\mathrm{fm}}^{\mathrm{df}}$ its restriction to disjunction free systems. These latter classes are the main focus of this paper.

A *configuration* of the counter system $S = \langle Q, \mathsf{X}_n, \Delta, \Lambda \rangle$ is a pair $(q, \boldsymbol{v}) \in Q \times \mathbb{Z}^n$, where q is the current control state and $\boldsymbol{v}[i]$ is the value of the counter x_i, for all $i \in [1, n]$. Given two configurations $\gamma = (q, \boldsymbol{v})$ and $\gamma' = (q', \boldsymbol{v}')$ and a transition rule δ, we write $\gamma \xrightarrow{\delta} \gamma'$ iff $q = \mathit{source}(\delta)$, $q' = \mathit{target}(\delta)$, $\boldsymbol{v} \models \mathit{guard}(\delta)$ and $\boldsymbol{v}' = \mathit{update}(\delta)(\boldsymbol{v})$. We use the notation $\gamma \to \gamma'$ when there exists a transition rule δ such that $\gamma \xrightarrow{\delta} \gamma'$. A *run* of S is then an infinite sequence of the form $\rho : \gamma_0 \xrightarrow{\delta_0} \gamma_1 \xrightarrow{\delta_1} \gamma_2 \xrightarrow{\delta_2} \ldots$. We say that such a run starts at configuration γ_0, furthermore we denote by $\mathit{trans}(\rho) = \delta_0\delta_1\delta_2\ldots$ the infinite sequence of transition rules seen during ρ. Without loss of generality we consider *deadlock-free* counter systems only, where for each configuration $\gamma \in Q \times \mathbb{Z}^n$, there exists a configuration γ' such that $\gamma \to \gamma'$[3].

Example 2. The sequence below is a run of the counter system from Figure 1:

$$\left(q_0, \begin{bmatrix}0\\0\\0\end{bmatrix}\right) \xrightarrow{\delta_0} \left(q_0, \begin{bmatrix}1\\0\\0\end{bmatrix}\right) \xrightarrow{\delta_1} \left(q_1, \begin{bmatrix}0\\0\\1\end{bmatrix}\right) \xrightarrow{\delta_2} \left(q_1, \begin{bmatrix}1\\1\\1\end{bmatrix}\right) \xrightarrow{\delta_3} \left(q_2, \begin{bmatrix}1\\1\\1\end{bmatrix}\right)$$

$$\xrightarrow{\delta_4} \left(q_2, \begin{bmatrix}2\\0\\1\end{bmatrix}\right) \xrightarrow{\delta_5} \left(q_3, \begin{bmatrix}2\\0\\1\end{bmatrix}\right) \xrightarrow{\delta_6} \left(q_3, \begin{bmatrix}2\\0\\1\end{bmatrix}\right) \xrightarrow{\delta_6} \left(q_3, \begin{bmatrix}2\\0\\1\end{bmatrix}\right) \xrightarrow{\delta_6} \cdots \quad \blacksquare$$

2.2 Decision Problems

The *reachability problem* for a class of counter systems $\mathcal{C}$, denoted by $\textsc{Reach}(\mathcal{C})$, can then be stated as follows: given a counter system S in $\mathcal{C}$, an initial configuration γ_0, and a control state q_f, does S have a run starting in γ_0 and containing a configuration $(q_f, \boldsymbol{v})$, for some $\boldsymbol{v} \in \mathbb{Z}^n$? It is well known that $\textsc{Reach}(\mathsf{TS})$ is undecidable for non-flat counter systems, even for only 2 counters with zero test guards, and increment/decrement updates [24].

In this work we also consider *model checking* problems for two specification logics, namely Past Linear Temporal Logic (PLTL) and First Order Logic (FO). The formulae of PLTL are defined by the grammar: $\phi ::= \mathsf{p} \mid \neg\phi \mid \phi \wedge \phi \mid \mathsf{X}\phi \mid \phi\mathsf{U}\phi \mid \mathsf{X}^{-1}\phi \mid \phi\mathsf{S}\phi$, where $\mathsf{p} \in \mathsf{AP}$. As usual, we consider the derived modal operators $\mathsf{F}\phi := \top\mathsf{U}\phi$ and $\mathsf{G}\phi := \neg\mathsf{F}\neg\phi$. Given a run ρ: $\gamma_0 \xrightarrow{\delta_0} \gamma_1 \xrightarrow{\delta_1} \gamma_2 \xrightarrow{\delta_2} \ldots$ of a counter system S and a PLTL formula ϕ, the semantics of PLTL is defined by an inductive forcing relation $\rho, i \models_{\mathsf{PLTL}} \phi$, where for all $i \geq 0$: $\rho, i \models_{\mathsf{PLTL}} \mathsf{p} \Leftrightarrow$

[3] We ensure deadlock-freedom by adding a sink state σ to S, with a self-loop $\sigma \xrightarrow{\top} \sigma$, and a transition $q \xrightarrow{\top} \sigma$ from each state $q \in Q$.

$\gamma_i = (q, \boldsymbol{v})$ and $\mathsf{p} \in \Lambda(q)$; $\rho, i \models_{\mathsf{PLTL}} \mathsf{X}\phi \Leftrightarrow \rho, i+1 \models_{\mathsf{PLTL}} \phi$; $\rho, i \models_{\mathsf{PLTL}} \phi \mathsf{U} \psi \Leftrightarrow \rho, j \models_{\mathsf{PLTL}} \psi$ for some $j \geq i$ and $\rho, k \models_{\mathsf{PLTL}} \phi$ for all $i \leq k < j$; $\rho, i \models_{\mathsf{PLTL}} \mathsf{X}^{-1}\phi \Leftrightarrow i > 0$ and $\rho, i-1 \models_{\mathsf{PLTL}} \phi$; $\rho, i \models_{\mathsf{PLTL}} \phi \mathsf{S} \psi \Leftrightarrow \rho, j \models_{\mathsf{PLTL}} \psi$ for some $0 \leq j \leq i$ and $\rho, k \models_{\mathsf{PLTL}} \phi$ for all $j < k \leq i$. The semantics of the boolean connectives $\wedge$ and $\neg$ is the usual one. We write $\rho \models_{\mathsf{PLTL}} \phi$ for $\rho, 0 \models_{\mathsf{PLTL}} \phi$. For instance, each run of the counter system from Figure 1 satisfies $\mathsf{G}((\mathsf{b} \wedge \mathsf{Xb} \wedge \mathsf{Fd}) \rightarrow \mathsf{F}(\mathsf{c} \wedge \mathsf{Xc}))$, because each run visiting q_3 sees the same number of b's and c's.

The formulae of FO are defined by the grammar: $\phi ::= \mathsf{p}(\mathsf{z}) \mid \mathsf{z} < \mathsf{z}' \mid \neg\phi \mid \phi \wedge \phi \mid \exists \mathsf{z}.\phi$, where $\mathsf{p} \in \mathsf{AP}$ and z belongs to a countable set of *logical variables* Var. The semantics is given by a forcing relation $\rho \models_{\mathsf{FO}} \phi$ between runs ρ of S and closed formulae ϕ, with no free variables, which interprets the quantified variables $\mathsf{z} \in \mathsf{Var}$ as positive integers denoting positions in the run. With this convention, the semantics of FO is standard. For instance, each run of the counter system from Figure 1 satisfies the FO property: $\forall \mathsf{x} \forall \mathsf{x}'.(\mathsf{x} < \mathsf{x}' \wedge \mathsf{b}(\mathsf{x}) \wedge \mathsf{b}(\mathsf{x}') \wedge \exists \mathsf{z}.\mathsf{d}(\mathsf{z})) \rightarrow \exists \mathsf{y} \exists \mathsf{y}' \,.\, \mathsf{c}(\mathsf{y}) \wedge \mathsf{c}(\mathsf{y}')$, which differs from the previous PLTL formula only in that x and x' (y and y') are not necessarily consecutive moments in time. For both of these logics, we consider the size of a formula as its number of subformulae.

The model-checking problem for counter systems in a class $\mathcal{C}$ with specification language $\mathcal{L}$ (in this work either PLTL or FO), denoted by $\mathrm{MC}_{\mathcal{L}}(\mathcal{C})$, is defined as follows: given a counter system S in $\mathcal{C}$, an initial configuration γ_0, and a formula ϕ of $\mathcal{L}$, does there exist a run ρ of S starting in γ_0 such that $\rho \models_{\mathcal{L}} \phi$.

Table 1. Known results

	KS	KS_{f}	TS	TS_{f}	$\mathsf{AS}_{\mathrm{fm}}$
Reach	NLOGSPACE	NLOGSPACE	Undec. [24]	NP-c.[10]	4EXPtime [12]
$\mathrm{MC}_{\mathsf{PLTL}}$	PSPACE-c.[25]	NP-c.[10,20]	Undec.	NP-c. [10]	4EXPTIME [11]
$\mathrm{MC}_{\mathsf{FO}}$	NONELEM. [26]	PSPACE-c. [8]	Undec.	PSPACE-c. [8]	Decid. [11]

Table 1 gives an overview of the known complexity bounds for the previously mentioned decision problems looking at different classes of counter systems. For flat Kripke structures, it is proved in [10,20] that $\mathrm{MC}_{\mathsf{PLTL}}(\mathsf{KS}_{\mathrm{f}})$ is NP-complete and in [8] that $\mathrm{MC}_{\mathsf{FO}}(\mathsf{KS}_{\mathrm{f}})$ is Pspace-complete, whereas $\mathrm{MC}_{\mathsf{PLTL}}(\mathsf{KS})$ is Pspace-complete and $\mathrm{MC}_{\mathsf{FO}}(\mathsf{KS})$ is non-elementary. As explained in [8,10], the complexity of these two last problems does not change if one considers flat translating counter systems. For what concerns flat counter systems with the finite monoid property, it has been shown that one can compute a Presburger formula which characterizes the reachability set, which entails the decidability of Reach($\mathsf{AS}_{\mathrm{fm}}$) [12]. Later on, in [11], the authors have shown that the model-checking of an extension of the branching time logic CTL^* is decidable. Hence we know that $\mathrm{MC}_{\mathsf{PLTL}}(\mathsf{AS}_{\mathrm{fm}})$ and $\mathrm{MC}_{\mathsf{FO}}(\mathsf{AS}_{\mathrm{fm}})$ are decidable, however no precise complexity for these problems is known. We only can deduce from the proofs in [12,5,11] that for Reach($\mathsf{AS}_{\mathrm{fm}}$) and $\mathrm{MC}_{\mathsf{PLTL}}(\mathsf{AS}_{\mathrm{fm}})$ there exists a reduction to the satisfiability problem for Presburger arithmetic where the built formula is

exponentially bigger than the size of the model, this leads to an upper bound in 4EXPTIME (the satisfiability problem for Presburger arithmetic can in fact be solved in 3EXPTIME, see e.g. [16]).

In this work, we aim at improving the complexity for the problems related to affine counter systems with the finite monoid property. Note that for the presented results, the counter systems were manipulating natural numbers instead of integers, but considering the latter option does not change the stated results.

3 A Hardness Result

In this section we prove that the reachability problem for flat affine counter systems with the finite monoid property is Σ_2^P-hard, by reduction from the validity problem for the $\exists^*\forall^*$ fragment of quantified boolean formulae (Σ_2-QBF), which is a well-known Σ_2^P-complete problem [1, §5.2]. Let us consider a formula $\Phi := \exists y_1 \dots \exists y_p \forall z_1 \dots \forall z_q \,.\, \Psi(\boldsymbol{y}, \boldsymbol{z})$, where $\boldsymbol{y} = \{y_1, \dots, y_p\}$ and $\boldsymbol{z} = \{z_1, \dots, z_q\}$ are non-empty sets of boolean variables, and Ψ is a quantifier-free boolean formula. We shall build, in polynomial time, a flat counter system S_Φ, with the finite monoid property, such that Φ is valid if and only if S_Φ has a run reaching q_f which starts in $(q_0, \boldsymbol{v}_0)$ for a certain valuation $\boldsymbol{v}_0$ of its counters.

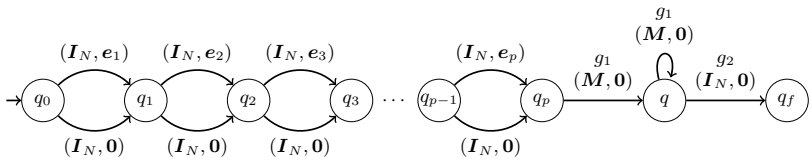

Fig. 2. The counter system S_Φ corresponding to the Σ_2-QBF Φ

Let π_n denote the n-th prime number, i.e. $\pi_1 = 2, \pi_2 = 3, \pi_3 = 5$, etc. Formally, $S_\Phi = \langle Q, \mathsf{X}_N, \Delta, \Lambda \rangle$, where $Q = \{q_0, \dots, q_p, q, q_f\}$, $N = p + \sum_{n=1}^{q} \pi_n$, and Λ is the function associating to each state an empty set of propositions. We recall that π_n is a polynomial in the size of n, hence N is as well polynomial in the size of n. The transition rules Δ are depicted in Figure 2. Intuitively, each existentially quantified boolean variable y_i of Φ is modeled by the counter x_i in S_Φ, each universally quantified variable z_j of Φ is modeled by the counter $x_{p+\sum_{n=1}^{j} \pi_n}$, and the rest are working counters. All counters range over the set $\{0, 1\}$, with the obvious meaning (0 stands for false and 1 for true).

The counter system S_Φ works in two phases. The first phase, corresponding to transitions $q_0 \to \dots \to q_p$, initializes the counters $x_1, \dots, x_p$ to some values from the set $\{0, 1\}$, thus mimicking a choice of boolean values for the existentially quantified variables $y_1, \dots, y_p$ from Φ. Here $\mathbf{I}_N \in \mathbb{Z}^{N \times N}$ is the identity matrix, and $\boldsymbol{e}_i \in \{0, 1\}^N$ is the unit vector such that $\boldsymbol{e}_i[j] = 0$ if $j \neq i$ and $\boldsymbol{e}_i[i] = 1$.

The second phase checks that Φ is valid for each choice of $z_1, \dots, z_q$. This is done by the cycle $q \to q$, which explores all combinations of 0's and 1's for

the counters $x_{p+\sum_{n=1}^{j}\pi_n}$, corresponding to z_j, for all $j \in [1,q]$. To this end, we use the permutation matrix $\boldsymbol{M}$, which consists of $\mathbf{I}_p$ and q rotation blocks $\boldsymbol{M}_{\pi_j} \in \{0,1\}^{\pi_j \times \pi_j}$ (Figure 3). The valuation $\boldsymbol{v}_0$ ensures that the initial value of $x_{p+\sum_{n=1}^{j}\pi_n}$ is 1, for all $j \in [1,q]$, the other counters being 0 initially (Figure 3).

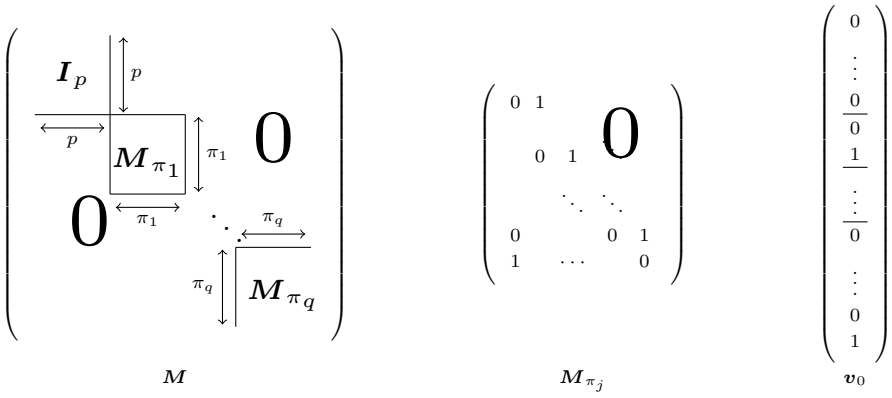

Fig. 3. Matrix $\boldsymbol{M}$ and initial vector $\boldsymbol{v}_0$

Intuitively, after n iterations of the affine function $(\boldsymbol{M}, \mathbf{0})$, labeling the cycle $q \to q$ in S_Φ, we have $x_{p+\sum_{n=1}^{j}\pi_n} = 1$ iff n is a multiple of π_j. This fact guarantees that all combinations of 0's and 1's for $z_1, \ldots, z_q$ have been visited in $\Pi_{j=1}^{q}\pi_j$ iterations of the cycle. The guard g_1, labeling the cycle, tests that, at each iteration, the formula Ψ is satisfied, using a standard encoding of the formula Ψ. Namely, each variable y_i is encoded as the term $x_i \geq 1$ and each z_j is encoded as $x_{p+\sum_{n=1}^{j}\pi_n} \geq 1$.

For instance, the formula $y_1 \vee \neg z_2$ is encoded as $x_1 \geq 1 \vee \neg(x_{p+\pi_1+\pi_2} \geq 1))$ which is equivalent to $x_1 \geq 1 \vee x_{p+\pi_1+\pi_2} < 1$. Finally, the guard g_2 simply checks that $x_{\pi_1} = \ldots = x_{\pi_1+\ldots+\pi_q} = 1$, ensuring that the loop has been iterated sufficiently many times. This allows us to deduce the following result.

Lemma 1. REACH($\mathsf{AS}_{\mathrm{fm}}$) *is* Σ_2^P*-hard.*

4 Bounding the Number of Cycle Iterations

In this section we prove a crucial property of counter systems from the $\mathsf{AS}_{\mathrm{fm}}^{\mathrm{df}}$ class, namely that there exists a polynomial function $\mathsf{Poly}(x)$ such that, for each run ρ starting at γ_0 of the considered counter system, there exists another run ρ' starting at γ_0, using the same transition rules as ρ, in exactly the same order, and which iterates each simple cycle at most $2^{\mathsf{Poly}(size(S)+size(\gamma_0))}$ times.

In the rest of this section, we fix a flat disjunction free affine counter system $S = \langle Q, \mathsf{X}_n, \Delta, \Lambda \rangle$ with the finite monoid property. We recall here that the set of

runs of a flat counter system can be captured by a finite (though exponential) number of *path schemas* [8]. Formally, a path schema is a non-empty finite sequence $P := u_1 \dots u_N$, where u_i is either a transition rule from Δ or a simple cycle, such that (i) $u_1, \dots, u_N$ are pairwise distinct, (ii) u_N is a simple cycle, called *terminal*, and (iii) $target(u_i) = source(u_{i+1})$, for all $i \in [1, N-1]$. All simple cycles on P, except for u_N, are called *nonterminal*. We use then the following notations: $len(P)$ for N, $P[i]$ for u_i with $i \in [1, N]$, and $size(P)$ is the sum of the sizes of all transition rules occurring in P.

Intuitively a path schema P represents a set of infinite paths obtained by iterating the non-terminal cycles a certain number of times. We can hence represent such a path by an associated path schema and an iteration vector. Formally, an *iterated path schema* is a pair $\langle P, \boldsymbol{m} \rangle$, such that P is a path schema, and $\boldsymbol{m} \in \mathbb{N}^{len(P)-1}$ is a vector, where for all $i \in [1, len(P)-1]$, $\boldsymbol{m}[i] \geq 1$ and $\boldsymbol{m}[i] > 1$ implies that $P[i]$ is a cycle. An iterated path schema defines a unique infinite word over Δ, denoted by $trans(P, \boldsymbol{m}) = P[1]^{\boldsymbol{m}[1]} P[2]^{\boldsymbol{m}[2]} \cdots P[len(P)-1]^{\boldsymbol{m}[len(P)-1]} P[len(P)]^{\omega}$. We recall the following result:

Lemma 2. *[10] Let S be a flat affine counter system. Then:*

1. *the length and the size of a path schema of S are polynomial in $size(S)$;*
2. *for any run ρ of S, there exists an iterated path schema $\langle P, \boldsymbol{m} \rangle$ such that $trans(\rho) = trans(P, \boldsymbol{m})$.*

For a run ρ, we consider the set $ips(\rho) = \{\langle P, \boldsymbol{m} \rangle \mid trans(\rho) = trans(P, \boldsymbol{m})\}$. Observe that $ips(\rho) \neq \emptyset$ for any run ρ of S, due to Lemma 2 (2). Moreover, as a consequence of Lemma 2 (1), the number of path schemas is bounded by a simple exponential in the size of S. Note that $ips(\rho)$ is not necessarily a singleton: if a run enters and exits a loop in different states then, in the path schema, the loop may begin either from the entering state or from the exiting state.

We fix a run ρ of S starting at γ_0, and $\langle P, \boldsymbol{m} \rangle \in ips(\rho)$ an iterated path schema corresponding to ρ. We consider a simple cycle $c = \delta_0 \dots \delta_{k-1}$ of P, whose transition rules are $\delta_i = \langle q_i, \boldsymbol{C}_i \cdot x \leq \boldsymbol{d}_i, (\boldsymbol{A}_i, \boldsymbol{b}_i), q_{i+1} \rangle$, for all $i \in [0, k-1]$, and $q_k = q_0$. Let $f_c = (\boldsymbol{A}_c, \boldsymbol{b}_c)$ be the update of the entire cycle c, where $\boldsymbol{A}_c = \boldsymbol{A}_{k-1} \cdots \boldsymbol{A}_1 \cdot \boldsymbol{A}_0$, denoted $\prod_{i=k-1}^{0} \boldsymbol{A}_i$, and $\boldsymbol{b}_c = \sum_{i=0}^{k-1} \prod_{j=k-1}^{i+1} \boldsymbol{A}_j \cdot \boldsymbol{b}_i$. Since S has the finite monoid property, the set $\mathcal{M}_c = \left\{ \boldsymbol{A}_c^i \mid i \in \mathbb{N} \right\}$ is finite. Then there exist two integer constants $\alpha, \beta \in \mathbb{N}$, such that $0 \leq \alpha + \beta \leq \|\mathcal{M}_c\| + 1$, and $\boldsymbol{A}_c^{\alpha} = \boldsymbol{A}_c^{\alpha+\beta}$. Observe that, in this case, we have $\mathcal{M}_c = \left\{ \boldsymbol{A}_c^0, \dots, \boldsymbol{A}_c^{\alpha}, \dots, \boldsymbol{A}_c^{\alpha+\beta-1} \right\}$.

Our goal is to exhibit another run ρ' of S and an iterated path schema $\langle P, \boldsymbol{m}' \rangle \in ips(\rho')$, such that $\| \boldsymbol{m}' \|_{\infty} \leq 2^{\mathsf{Poly}(size(S)+size(\gamma_0))}$, for a polynomial function $\mathsf{Poly}(x)$. Because $c = \delta_0 \dots \delta_{k-1}$ is a simple cycle of P and $\langle P, \boldsymbol{m} \rangle \in ips(\rho)$, there exists a (possibly infinite) contiguous subsequence of ρ, let us call it $\theta = (q_0, \boldsymbol{v}_0) \xrightarrow{\tau_0} (q_1, \boldsymbol{v}_1) \xrightarrow{\tau_1} \dots$ that iterates c, i.e. $\tau_i = \delta_{(i \bmod k)}$, for all $i \geq 0$. In the following, we call any subsequence of a run an *execution*.

The main intuition now is that θ can be decomposed into a prefix of length $(\alpha + \beta)k$ and k infinite sequences of translations along some effectively computable vectors $\boldsymbol{w}_0, \dots, \boldsymbol{w}_{k-1}$. More precisely, all valuations $\boldsymbol{v}_i$ of θ, for $i \geq$

$(\alpha + \beta)k$, that are situated at distance βk one from another, differ by exactly the same vector. We refer to Figure 4 for an illustration of this idea.

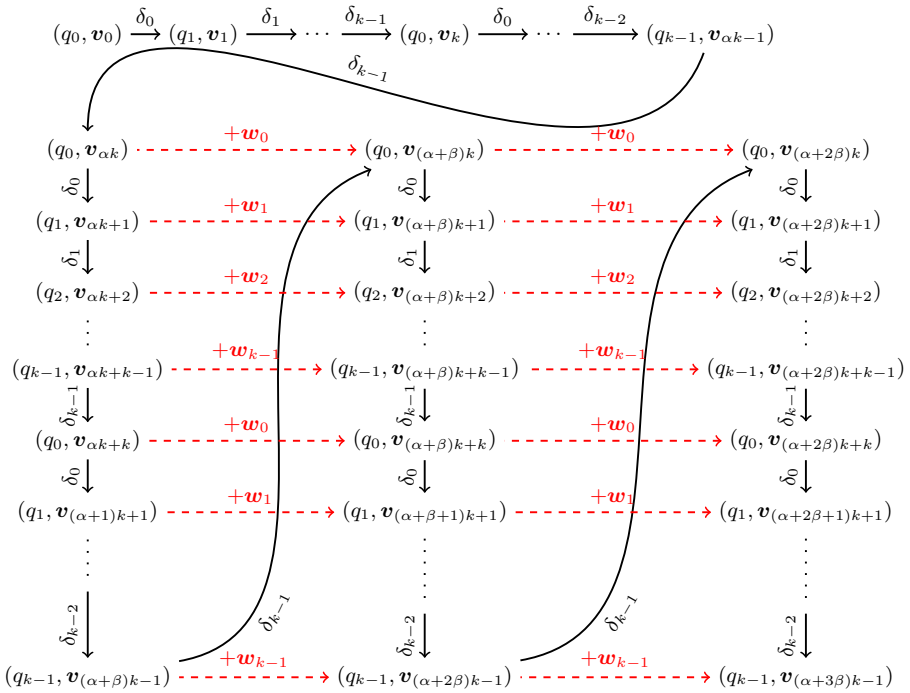

Fig. 4. Behavior of an execution which iterates $\alpha + 3\beta$ times the cycle $c = \delta_0 \dots \delta_{k-1}$

Lemma 3. *Given an execution* $(q_0, \boldsymbol{v}_0) \xrightarrow{\delta_0} \dots \xrightarrow{\delta_{k-1}} (q_k, \boldsymbol{v}_k) \xrightarrow{\delta_0} \dots$ *of* S *that iterates a simple cycle* $c = \delta_0 \dots \delta_{k-1}$*, there exist* $\boldsymbol{w}_0, \dots, \boldsymbol{w}_{k-1} \in \mathbb{Z}^n$*, such that* $\boldsymbol{v}_{(\alpha+p\beta+r)k+q} = \boldsymbol{v}_{(\alpha+r)k+q} + p \cdot \boldsymbol{w}_q$*, for all* $p \geq 0, r \in [0, \beta - 1]$ *and* $q \in [0, k-1]$*, where* $f_c = (\boldsymbol{A}_c, \boldsymbol{b}_c)$ *is the update of* c *and* $\alpha, \beta \geq 0$ *are such that* $\boldsymbol{A}_c^\alpha = \boldsymbol{A}_c^{\alpha+\beta}$*.*

We distinguish now the case when c is a nonterminal cycle of P, iterated finitely many times, from the case when c is terminal, thus iterated ad infinitum. We consider first the case when c is a nonterminal cycle, taken a finite number of times. Viewing the sequence of counter valuations, that occur during the unfolding of a simple loop, as a set of translations by vectors $\boldsymbol{w}_0, \dots, \boldsymbol{w}_{k-1}$, prefixed by an initial sequence, allows us to reduce the problem of checking the validity of the guards along this sequence to checking the guards only in the beginning and in the end of each translation by $\boldsymbol{w}_q$, for $q \in [0, k-1]$. This is possible because the counter systems is disjunction free and hence each guard in the loop is defined by a convex vector set $\{\boldsymbol{v} \in \mathbb{Z}^n \mid \boldsymbol{C} \cdot \boldsymbol{v} \leq \boldsymbol{d}\}$, for a matrix $\boldsymbol{C} \in \mathbb{Z}^{m \times n}$ and a vector $\boldsymbol{d} \in \mathbb{Z}^m$, thus a sequence of vectors produced by a

translation cannot exit and then re-enter the same guard, later on. This crucial observation, needed to prove the upper bound, is formalized below.

We consider the relaxed transition relation $\rightsquigarrow \subseteq (Q \times \mathbb{Z}^n) \times \Delta \times (Q \times \mathbb{Z}^n)$, defined as $(q, \boldsymbol{v}) \overset{\delta}{\rightsquigarrow} (q', \boldsymbol{v}')$ iff $\mathit{source}(\delta) = q$, $\boldsymbol{v}' = \mathit{update}(\delta)(\boldsymbol{v})$ and $\mathit{target}(\delta) = q'$. Hence, $\rightsquigarrow$ allows to move from one configuration to another as in $\rightarrow$, but without testing the guards. In the following, we fix a sequence of configurations $\theta' = (q_0, \boldsymbol{v}_0) \overset{\tau_0}{\rightsquigarrow} (q_1, \boldsymbol{v}_1) \overset{\tau_1}{\rightsquigarrow} \dots$ called a *pseudo-execution*. We assume, moreover, that θ' iterates the simple cycle $c = \delta_0, \dots, \delta_{k-1}$ a finite number of times, i.e. $\tau_i := \delta_{i \bmod k}$, for all $i \geq 0$. To check whether θ' is a real execution, it is enough to check the guards in the first $\alpha + \beta + 1$ and the last β iterations of the cycle, as shown by the following lemma:

Lemma 4. *For any $m > (\alpha+\beta+1)k$, given a finite pseudo-execution $(q_0, \boldsymbol{v}_0) \overset{\tau_0}{\rightsquigarrow} \dots \overset{\tau_{m-1}}{\rightsquigarrow} (q_m, \boldsymbol{v}_m)$ of S, that iterates a nonterminal simple cycle $c = \delta_0 \dots \delta_{k-1}$, $(q_0, \boldsymbol{v}_0) \xrightarrow{\tau_0} \dots \xrightarrow{\tau_{m-1}} (q_m, \boldsymbol{v}_m)$ is an execution of S iff $\boldsymbol{v}_i \models \mathit{guard}(\tau_i)$, for all $i \in [0, (\alpha + \beta + 1)k - 1] \cup [m - \beta k, m - 1]$.*

The next step is to show that if a cycle is iterated ℓ times with $\ell = \alpha + \beta + p\beta + r$ for some $p > 0$ and $r \in [0, \beta - 1]$, starting with values $\boldsymbol{v} \in \mathbb{Z}^n$, then $[\boldsymbol{v}[1], \dots, \boldsymbol{v}[n], p]^\top$ is the solution of a system of inequations $\boldsymbol{M}_c \cdot [\boldsymbol{y}; z]^\top \leq \boldsymbol{n}_c$, where $[\boldsymbol{y}; z] = [y_1, \dots, y_n, z]$ is a vector of $n + 1$ variables. The bound on the number of iterations follows from the theorem below, by proving that the sizes of the entries of $\boldsymbol{M}_c$ and $\boldsymbol{n}_c$ (in binary) are bounded by a polynomial in $\mathit{size}(S)$.

Theorem 1. *Given $\boldsymbol{A} \in \mathbb{Z}^{m \times n}$ and $\boldsymbol{b} \in \mathbb{Z}^m$, for $n \geq 2$, the system $\boldsymbol{A} \cdot \boldsymbol{x} \leq \boldsymbol{b}$ has a solution in $\mathbb{N}^n$ iff it has a solution such that $\|\boldsymbol{x}\|_\infty \leq m^{2n} \cdot \|\boldsymbol{A}\|_{\max}^n \cdot \|\boldsymbol{b}\|_\infty$.*

We recall that $c = \delta_0, \dots, \delta_{k-1}$, where $\mathit{guard}(\delta_i) := \boldsymbol{C}_i \cdot \boldsymbol{x} \leq \boldsymbol{d}_i$, $\mathit{update}(\delta_i) := (\boldsymbol{A}_i, \boldsymbol{b}_i)$, and that $f_c = (\boldsymbol{A}_c, \boldsymbol{b}_c)$ is the affine function defining the update of the entire cycle. For any $j > 0$, we define $\boldsymbol{b}_c^j = \Sigma_{i=0}^{j-1} \boldsymbol{A}_c^i \cdot \boldsymbol{b}_c$, hence $f_c^\ell = (\boldsymbol{A}_c^\ell, \boldsymbol{b}_c^\ell)$ is the update corresponding to ℓ iterations of the cycle for a fixed integer constant $\ell > 0$. The following set of inequalities expresses the fact that all guards are satisfied within the ℓ-th iteration of the cycle starting at $\boldsymbol{v} \in \mathbb{Z}^n$:

$$\boldsymbol{C}_p \cdot \Big(\prod_{i=p-1}^{0} \boldsymbol{A}_i \cdot (\boldsymbol{A}_c^{\ell-1} \cdot \boldsymbol{v} + \boldsymbol{b}_c^{\ell-1}) \Big) + \sum_{i=0}^{p-1} \boldsymbol{A}_{p-1} \cdots \boldsymbol{A}_{i+1} \cdot \boldsymbol{b}_i \Big) \leq \boldsymbol{d}_p, \text{ for all } p = 0, \dots, k-1$$

In the sequel, we define $\boldsymbol{M}_\ell$ as the matrix obtained by vertically stacking the matrices $\boldsymbol{C}_j \cdot \prod_{i=j-1}^{0} \boldsymbol{A}_i \cdot \boldsymbol{A}_c^{\ell-1}$ for $j = 0, \dots, k-1$, with $\boldsymbol{C}_0 \cdot \boldsymbol{A}_c^{\ell-1}$ on top. Also, $\boldsymbol{n}_\ell$ is the column vector with rows $\boldsymbol{n}_\ell[j] = \boldsymbol{d}_j - \big(\boldsymbol{C}_j \cdot \prod_{i=j-1}^{0} \boldsymbol{A}_i \cdot \boldsymbol{b}_c^{\ell-1} + \boldsymbol{C}_j \cdot \big(\sum_{i=0}^{j-1} \boldsymbol{A}_{j-1} \cdots \boldsymbol{A}_{i+1} \cdot \boldsymbol{b}_i \big) \big)$, for $j = 0, \dots, k-1$. For technical reasons that will be made clear next, we do not need to consider the case when the loop is iterated less than $\alpha + 2\beta + 1$ times. We know, from Lemma 4, that checking whether a given cycle c can be iterated $\ell > \alpha + 2\beta + 1$ times from $\boldsymbol{v}$, reduces to checking

the validity of the guards during the first $\alpha+\beta+1$ and the last β iterations only. This condition is encoded by the union of the linear inequality systems below:

$$\begin{bmatrix} \boldsymbol{M}_1 \\ \cdots \\ \boldsymbol{M}_{\alpha+\beta+1} \end{bmatrix} \cdot \boldsymbol{v} \leq \begin{bmatrix} \boldsymbol{n}_1 \\ \cdots \\ \boldsymbol{n}_{\alpha+\beta+1} \end{bmatrix} \qquad \begin{bmatrix} \boldsymbol{M}_1 \\ \cdots \\ \boldsymbol{M}_{\beta} \end{bmatrix} \cdot f_c^{\ell-\beta}(\boldsymbol{v}) \leq \begin{bmatrix} \boldsymbol{n}_1 \\ \cdots \\ \boldsymbol{n}_{\beta} \end{bmatrix}$$

Since we assumed that $\ell > \alpha+2\beta+1$, it follows that $\ell-\beta = \alpha+p\beta+r$ for some $p > 0$ and $r \in [0, \beta-1]$, thus $f_c^{\ell-\beta}(\boldsymbol{v}) = f_c^{\alpha+r}(\boldsymbol{v})+p\cdot\boldsymbol{w}_0 = \boldsymbol{A}_c^{\alpha+r}\cdot\boldsymbol{v}+\boldsymbol{b}_c^{\alpha+r}+p\cdot\boldsymbol{w}_0$, by Lemma 3. Then, for any finite execution starting with $\boldsymbol{v}$, and consisting of $\alpha+p\beta+r$ iterations of c, we have that the column vector $[\boldsymbol{v}[1], \ldots, \boldsymbol{v}[n], p]^\top$ is a solution of the linear system $\boldsymbol{M}_{c,r} \cdot [\boldsymbol{y}; z]^\top \leq \boldsymbol{n}_{c,r}$, where:

$$\boldsymbol{M}_{c,r} = \begin{bmatrix} \boldsymbol{M}_1 & & \boldsymbol{0} \\ & \cdots & \\ \boldsymbol{M}_{\alpha+\beta+1} & & \boldsymbol{0} \\ \boldsymbol{M}_1 \cdot \boldsymbol{A}_c^{\alpha+r} & & \boldsymbol{M}_1 \cdot \boldsymbol{w}_0 \\ & \cdots & \\ \boldsymbol{M}_\beta \cdot \boldsymbol{A}_c^{\alpha+r} & & \boldsymbol{M}_\beta \cdot \boldsymbol{w}_0 \end{bmatrix} \qquad \boldsymbol{n}_{c,r} = \begin{bmatrix} \boldsymbol{n}_1 \\ \cdots \\ \boldsymbol{n}_{\alpha+\beta+1} \\ \boldsymbol{n}_1 - \boldsymbol{M}_1 \cdot \boldsymbol{b}_c^{\alpha+r} \\ \cdots \\ \boldsymbol{n}_\beta - \boldsymbol{M}_\beta \cdot \boldsymbol{b}_c^{\alpha+r} \end{bmatrix}$$

We now consider the case when the simple cycle $c = \delta_0 \ldots \delta_{k-1}$ is terminal and let $\boldsymbol{w}_0, \ldots, \boldsymbol{w}_{k-1} \in \mathbb{Z}^n$ be the vectors from Lemma 3. We say that c is *infinitely iterable* iff for all $i \in [0, k-1]$, we have $\boldsymbol{C}_i \cdot \boldsymbol{w}_i \leq 0$. Since $\boldsymbol{w}_0, \ldots, \boldsymbol{w}_{k-1}$ are effectively computable vectors[4], this condition is effective. The next lemma reduces the existence of an infinite iteration of the cycle to the existence of an integer solution of a linear inequation system.

Lemma 5. *Given an infinite pseudo-execution* $(q_0, \boldsymbol{v}_0) \overset{\tau_0}{\rightsquigarrow} (q_1, \boldsymbol{v}_1) \overset{\tau_1}{\rightsquigarrow} \ldots$ *of* S*, that iterates a terminal simple cycle* $c = \delta_0 \ldots \delta_{k-1}$*,* $(q_0, \boldsymbol{v}_0) \xrightarrow{\tau_0} (q_1, \boldsymbol{v}_1) \xrightarrow{\tau_1} \ldots$ *is an infinite execution of* S *iff* c *is infinitely iterable and* $\boldsymbol{v}_i \models \mathit{guard}(\tau_i)$*, for all* $i \in [0, (\alpha+\beta+1)k-1]$*.*

As a consequence, for an infinitely iterable cycle c, the existence of an execution that iterates c infinitely often is captured by the linear system $\boldsymbol{M}_{c,\omega} \cdot \boldsymbol{y} \leq \boldsymbol{n}_{c,\omega}$, where $\boldsymbol{M}_{c,\omega}$ and $\boldsymbol{n}_{c,\omega}$ are obtained by stacking the matrices $\boldsymbol{M}_1, \ldots, \boldsymbol{M}_{\alpha+\beta+1}$ and vectors $\boldsymbol{n}_1, \ldots, \boldsymbol{n}_{\alpha+\beta+1}$, respectively.

We have now all the ingredients needed to bound the number of cycle iterations within the runs of a flat disjunction free affine counter system having the finite monoid property. The argument used in the proof relies on the result of Theorem 1, namely that the size of a minimal solution of a linear system of inequalities is polynomially bounded in the maximum absolute value of its coefficients, and the number of rows, and exponentially bounded in the number of columns. Since the number of rows depends on the maximum size of the monoids of the update matrices in the counter system, we use the result from [18, Lemma 13, §B.1], namely that the size of a finite monoid of a square matrix is simply exponential in the dimension of that matrix.

[4] They are defined in the proof of Lemma 3.

Theorem 2. *Given a flat disjunction free affine counter system $S = \langle Q, \mathsf{X}_n, \Delta, \Lambda\rangle$, with the finite monoid property, for any run ρ of S, starting in $(q_0, \boldsymbol{v}_0)$, and any iterated path schema $\langle P, \boldsymbol{m}\rangle \in ips(\rho)$, there exists a run ρ', starting in $(q_0, \boldsymbol{v}_0)$, and an iterated path schema $\langle P, \boldsymbol{m}'\rangle \in ips(\rho')$, such that $\| \boldsymbol{m}' \|_\infty \leq 2^{\mathsf{Poly}(size(S)+size(\boldsymbol{v}_0))}$, for a polynomial function $\mathsf{Poly}(x)$.*

5 The Complexities of Decision Problems for $\mathbf{AS}^{\mathbf{df}}_{\mathbf{fm}}$

In this section, we will prove that the previous reasoning on iterated path schemas allows us to deduce complexity bounds of the reachability problems and of model-checking with PLTL and FO formulae for disjunction free flat counter systems with the finite monoid property.

5.1 Reachability is Σ_2^P

In this section we give the first upper bound, for the reachability problem and show that $\textsc{Reach}(\mathsf{AS}^{\mathrm{df}}_{\mathrm{fm}})$ is Σ_2^P. Even if this upper bound holds only for disjunction free counter system, we believe we could extend it to all the class $\mathsf{AS}_{\mathrm{fm}}$ by adapting the method presented in [10] to eliminate the disjunctions. This would allow us to match the lower bound from Section 3. However we did not wish to enter into the heavy details of eliminating disjunctions in this work, in order to focus more on the specific aspects of affine counter systems. Anyway the provided result improves the 4EXPTIME upper bound from Table 1. The crux of the proof is based on the result provided by Theorem 2 and it follows the following reasoning: we use a polynomial-time bounded nondeterministic Turing machine that guesses an iterated path schema and then a NP oracle to check whether a guard has been violated. This gives us an $\mathrm{NP}^{\mathrm{NP}}$ algorithm for $\textsc{Reach}(\mathsf{AS}^{\mathrm{df}}_{\mathrm{fm}})$, which then lies in Σ_2^P. Theorem 2 ensures us the soundness of the Algorithm and the correctness is provided by the fact that if, in an iterated path schema, no guard is violated then it corresponds necessarily to a run.

Let us now explain how our NP oracle works. The next lemma is based on the fact that any power $\boldsymbol{A}^k$ of a finite monoid matrix $\boldsymbol{A}$ can be computed in time polynomial in $size(\boldsymbol{A})$ and $\log_2 k$, using matrix exponentiation by squaring. The reason is that the value of an entry of any power of a finite monoid matrix $\boldsymbol{A}$ is bounded by an exponential in $size(\boldsymbol{A})$, thus the size of its binary representation is polynomially bounded by $size(\boldsymbol{A})$, and each step of the squaring algorithm takes polynomial time [18, Lemma 14, §B.1].

Lemma 6. *Given an iterated path schema $\langle P, \boldsymbol{m}\rangle$ of a counter system with the finite monoid property S and an initial configuration γ_0, checking whether there is no run ρ starting at γ_0 such that $\langle P, \boldsymbol{m}\rangle \in ips(\rho)$ is in* NP.

The next theorem gives the main result of this section.

Theorem 3. $\textsc{Reach}(\mathsf{AS}^{\mathrm{df}}_{\mathrm{fm}})$ *is* Σ_2^P.

5.2 PLTL Model Checking is Σ_2^P

For a PLTL formula ϕ, its temporal depth $td(\phi)$ is defined as the maximal nesting depth of temporal operators in ϕ, and the size of ϕ is its number of subformulae. In [10, Theorem 4.1], the authors have proved a *stuttering* theorem for PLTL stating that if an ω-word $w = w_1 w_2^M w_3$ over the alphabet 2^{AP} with $w_2 \neq \epsilon$ satisfies a PLTL formula ϕ (i.e. $w, 0 \models_{\mathsf{PLTL}} \phi$) and if $M \geq 2td(\phi) + 5$ then all ω-words $w' = w_1 w_2^{M'} w_3$ with $M' \geq 2td(\phi) + 5$ are such that $w', 0 \models_{\mathsf{PLTL}} \phi$. In other words, to verify if an ω-word with some repeated infix words satisfies a PLTL formula it is enough to verify the property for the ω-words where each infix is repeated at most $2td(\phi)+5$ times. This allows to deduce that the model-checking of PLTL for flat translating counter systems is NP-complete. We rewrite now in our terminology the main proposition which leads to this result.

In the sequel we consider a flat disjunction free counter system $S = \langle Q, \mathsf{X}_n, \Delta, \Lambda \rangle$ with the finite monoid property. For a finite sequence of transitions $\delta_1 \dots d_k$, we denote by $\Lambda(\delta_1 \dots \delta_k) = \Lambda(source(\delta_1)) \dots \Lambda(source(\delta_k))$ the finite word labeling the sequence with sets of atomic propositions. We lift this definition to iterated path schemas $\langle P, \boldsymbol{m} \rangle$ as $\Lambda(P, \boldsymbol{m}) = \Lambda(P[1])^{\boldsymbol{m}[1]} \Lambda(P[2])^{\boldsymbol{m}[2]} \cdots \Lambda(P[len(P) - 1])^{\boldsymbol{m}[len(P)-1]} \Lambda(P[len(P)])^{\omega}$. Observe that, for a run ρ of a counter system, if $\langle P, \boldsymbol{m} \rangle \in ips(\rho)$ is an iterated path schema, we have by definition of the semantics of PLTL that $\rho \models_{\mathsf{PLTL}} \phi$ iff $\Lambda(P, \boldsymbol{m}), 0 \models_{\mathsf{PLTL}} \phi$[5] for all PLTL formulae ϕ. Moreover, for each $m \in \mathbb{N}$, we define the function ξ_m mapping each vector $\boldsymbol{v} \in \mathbb{N}^k$ to $\xi_m(\boldsymbol{v}) \in \mathbb{N}^k$, where, for all $i \in [1, k]$: $\xi_m(\boldsymbol{v})[i] = \boldsymbol{v}[i]$ if $\boldsymbol{v}[i] < m$ and $\xi_m(\boldsymbol{v})[i] = m$ otherwise. Let us now recall the main technical propositions established in [10], which are a consequence of the stuttering theorem for PLTL and of the result on the complexity of model-checking ultimately periodic path with PLTL given in [23].

Lemma 7. *Let $\langle P, \boldsymbol{m} \rangle$ be an iterated path schema and ϕ a PLTL formula, then:*

1. *[10, Proposition 5.1] $\Lambda(P, \boldsymbol{m}), 0 \models_{\mathsf{PLTL}} \phi$ iff $\Lambda(P, \xi_{2td(\phi)+5}(\boldsymbol{m})), 0 \models_{\mathsf{PLTL}} \phi$,*
2. *[23, Theorem 3.2] Given finite words u and v, checking $uv^{\omega}, 0 \models_{\mathsf{PLTL}} \phi$ can be done in time polynomial in the sizes of uv and ϕ.*

We need furthermore a version of Theorem 2 above, which ensures that given an iterated path schema and a PLTL formula ϕ, we do not change the number of times a loop is iterated if this one is less than $2.td(\phi) + 5$. The proof of the next result can in fact be deduced by adapting the proof of Theorem 2 by unfolding the loop which are iterated less than $2.td(\phi) + 5$ for a given formula ϕ. As a consequence of Lemma 7, the new run ρ', obtained in the next lemma, is such that $\rho \models_{\mathsf{PLTL}} \phi$ iff $\rho' \models_{\mathsf{PLTL}} \phi$ for the considered PLTL formula ϕ.

Lemma 8. *For a run ρ of S starting in $(q_0, \boldsymbol{v}_0)$, an iterated path schema $\langle P, \boldsymbol{m} \rangle \in ips(\rho)$ and a PLTL formula ϕ, there exists a run ρ' starting in $(q_0, \boldsymbol{v}_0)$, and an iterated path schema $\langle P, \boldsymbol{m'} \rangle \in ips(\rho')$, such that $\|\boldsymbol{m'}\|_{\infty} \leq 2^{\mathsf{Poly}(size(S)+size(\boldsymbol{v}_0)+td(\phi))}$ for a polynomial $\mathsf{Poly}(x)$ and $\xi_{2td(\phi)+5}(\boldsymbol{m}) = \xi_{2td(\phi)+5}(\boldsymbol{m'})$.*

[5] We take here the classical semantics of PLTL over infinite words.

We can now explain why the model-checking of flat counter systems with the finite monoid property with PLTL formulae is in Σ_2^P. Given a flat counter system S with the finite monoid property, an initial configuration γ_0, and a PLTL formula ϕ, we guess an iterated path schema $\langle P, \boldsymbol{m}\rangle$ of polynomial size in the size of S, γ_0 and ϕ and we check whether $\Lambda(P, \xi_{2td(\phi)+5}(\boldsymbol{m})), 0 \models_{\mathsf{PLTL}} \phi$. This check can be done in polynomial time in the size of P and ϕ thanks to Lemma 7. Finally, we use the NP algorithm of Lemma 6 to verify that there exists a run ρ starting at γ_0, such that $\langle P, \boldsymbol{m}\rangle \in ips(\rho)$. This gives us a Σ_2^P algorithm whose correctness is ensured by Lemma 8 and Lemma 2.

Theorem 4. $\mathrm{MC}_{\mathsf{PLTL}}(\mathsf{AS}_{\mathrm{fm}}^{\mathrm{df}})$ *is* Σ_2^P.

5.3 FO Model Checking is PSPACE-complete

For a FO formula ϕ, its quantifier height $qh(\phi)$ is the maximal nesting depth of its quantifiers, and the size of ϕ is its number of subformulae. Similarly, as for the PLTL case, in [8, Theorem 6], a stuttering theorem for FO is provided, which says that that two ω-words $w = w_1 w_2^M w_3$ and $w = w_1 w_2^{M'} w_3$ with $w \neq \epsilon$ are indistinguishable by a FO formula ϕ if M and M' are strictly bigger than $2^{qh(\phi)+2}$. The main difference with PLTL is that this provides an exponential bound in the maximum number of times an infix of an ω-word needs to be repeated to satisfy a FO formula. In the sequel we consider a flat counter system $S = \langle Q, \mathsf{X}_n, \Delta, \Lambda\rangle$ with the finite monoid property and we reuse the notations introduced in the previous section. The results of [8] can be restated as follows.

Lemma 9. *Given an iterated path schema* $\langle P, \boldsymbol{m}\rangle$ *and a* FO *formula* ϕ*, then:*

1. *[8, Lemma 7]* $\Lambda(P, \boldsymbol{m}) \models_{\mathsf{FO}} \phi$ *iff* $\Lambda(P, \xi_{2^{qh(\phi)+2}}(\boldsymbol{m})) \models_{\mathsf{FO}} \phi$*,*
2. *[8, Theorem 9] Checking* $\Lambda(P, \boldsymbol{m}), 0 \models_{\mathsf{FO}} \phi$ *can be done in space polynomial in the sizes of* $\langle P, \boldsymbol{m}\rangle$ *and* ϕ*.*

As for the PLTL case, this allows us to deduce a NPSPACE algorithm for the model-checking problem of flat counter system with the finite monoid property with FO formulae. Since the problem is already PSPACE-hard for flat translating counter systems [8, Theorem 9], we conclude by the following theorem.

Theorem 5. $\mathrm{MC}_{\mathsf{FO}}(\mathsf{AS}_{\mathrm{fm}}^{\mathrm{df}})$ *is* PSPACE*-complete.*

References

1. Arora, S., Barak, B.: Computational complexity: a modern approach. Cambridge University Press (2009)
2. Bardin, S., Finkel, A., Petrucci, J.L.L.: Fast: Fast acceleration of symbolic transition systems. `http://tapas.labri.fr/trac/wiki/FASTer`
3. Blondin, M., Finkel, A., Göller, S., Haase, C., McKenzie, P.: Reachability in two-dimensional vector addition systems with states is pspace-complete. CoRR abs/1412.4259 (2014), `http://arxiv.org/abs/1412.4259`

4. Boigelot, B.: Symbolic Methods for Exploring Infinite State Spaces. PhD, Univ. de Liège (1999)
5. Bozga, M., Iosif, R., Konecný, F.: Deciding conditional termination. Logical Methods in Computer Science 10(3) (2014)
6. Bozga, M., Iosif, R., Konečný, F.: Fast acceleration of ultimately periodic relations. In: CAV. LNCS, vol. 6174, pp. 227–242 (2010)
7. Comon, H., Jurski, Y.: Timed automata and the theory of real numbers. In: CONCUR '99, Proceedings. pp. 242–257 (1999)
8. Demri, S., Dhar, A.K., Sangnier, A.: On the complexity of verifying regular properties on flat counter systems,. In: ICALP'13. LNCS, vol. 7966, pp. 162–173 (2013)
9. Demri, S., Dhar, A.K., Sangnier, A.: Equivalence between model-checking flat counter systems and presburger arithmetic. In: RP'14. LNCS, vol. 8762, pp. 85–97. Springer (2014)
10. Demri, S., Dhar, A.K., Sangnier, A.: Taming past LTL and flat counter systems. Inf. Comput. 242, 306–339 (2015)
11. Demri, S., Finkel, A., Goranko, V., van Drimmelen, G.: Model-checking CTL* over flat presburger counter systems. Journal of Applied Non-Classical Logics 20(4), 313–344 (2010)
12. Finkel, A., Leroux, J.: How to compose presburger-accelerations: Applications to broadcast protocols. In: FST TCS '02. pp. 145–156 (2002)
13. Gawlitza, T.M., Monniaux, D.: Invariant generation through strategy iteration in succinctly represented control flow graphs. LMCS 8(3) (2012)
14. Göller, S., Haase, C., Ouaknine, J., Worrell, J.: Model checking succinct and parametric one-counter automata. In: ICALP'10. LNCS, vol. 6199, pp. 575–586. Springer (2010)
15. Gurari, E.M., Ibarra, O.H.: The complexity of decision problems for finite-turn multicounter machines. J. Computer and System Sciences 22, 220–229 (1981)
16. Haase, C.: Subclasses of Presburger arithmetic and the weak EXP hierarchy. In: CSL-LICS '14. pp. 47:1–47:10. ACM (2014)
17. Hojjat, H., Iosif, R., Konečný, F., Kuncak, V., Rümmer, P.: Accelerating Interpolants, pp. 187–202. Springer Berlin Heidelberg (2012)
18. Iosif, R., Sangnier, A.: How hard is it to verify flat affine counter systems with the finite monoid property ? CoRR abs/1605.05836 (2016), `http://arxiv.org/abs/1605.05836`
19. Konecny, F., Iosif, R., Bozga, M.: Flata: a verification toolset for counter machines. `http://nts.imag.fr/index.php/Flata` (2009)
20. Kuhtz, L., Finkbeiner, B.: Weak Kripke structures and LTL. In: CONCUR'11. LNCS, vol. 6901, pp. 419–433. Springer (2011)
21. Leroux, J., Sutre, G.: On Flatness for 2-Dimensional Vector Addition Systems with States, pp. 402–416 (2004)
22. Lipton, R.J.: The reachability problem is exponential-space-hard. Tech. Rep. 62, Yale University, Department of Computer Science (1976)
23. Markey, N., Schnoebelen, P.: Model checking a path. In: CONCUR'03. LNCS, vol. 2761, pp. 248–262 (2003)
24. Minsky, M.: Computation: Finite and Infinite Machines. Prentice-Hall (1967)
25. Sistla, A., Clarke, E.: The complexity of propositional linear temporal logic. J. ACM 32(3), 733–749 (1985)
26. Stockmeyer, L.J.: The complexity of decision problems in automata and logic. Ph.D. thesis, MIT (1974)