Multiple counters automata, safety analysis and Presburger arithmetic

Hubert Comon and Yan Jurski

LSV, ENS Cachan
61 av. président Wilson
94235 Cachan cedex
France
E-mail {comon,jurski}@lsv.ens-cachan.fr

Abstract. We consider automata with counters whose values are updated according to signals sent by the environment. A transition can be fired only if the values of the counters satisfy some guards (the guards of the transition). We consider guards of the form $y_i \# y_j + c_{i,j}$ where y_i is either x'_i or x_i, the values of the counter i respectively after and before the transition, and $\#$ is any relational symbol in $\{=, \leq, \geq, >, <\}$. We show that the set of possible counter values which can be reached after any number of iterations of a loop is definable in the additive theory of $\mathbb{N}$ (or $\mathbb{Z}$ or $\mathbb{R}$ depending on the type of the counters). This result can be used for the safety analysis of multiple counters automata.

1 Introduction

Finite state automata provide a nice framework for the verification of reactive systems. Their main advantage is the equivalence between recognizability and definability in some decidable logic (e.g. Monadic Second Order Logic or some of its fragments such as temporal logics). This allows to verify fully automatically that some structure defined by an automaton satisfies a given formula. Automata techniques are even optimal for the model checking of temporal formulas (see [2]). The counterpart of these nice properties is the relatively weak expressiveness of the finite state automata models. Many actual reactive systems require additional data structures in order to be described in an accurate way. Many models extending finite automata have been introduced in the literature. The most well-known one is probably timed automata [1] which allow to consider some "real time" constraints while keeping the nice decidability properties (with a higher complexity).

One of the most important purpose of verification is the so-called *safety analysis* which reduces most of the time to the following question: "is a bad state reachable from the initial configuration ?" For finite state automata, it is not difficult to compute all reachable states. This is however a more delicate question with infinite states systems: the computability of reachable configurations depends on the model under consideration. Here, we aim at contributing to this question by giving some decidability results. The model we consider is a "multiple counters automaton". A configuration is not only described by a state of the system, but also by the values of finitely many *counters* which may take arbitrary (integer or real) values. Such counters (also called *clocks* in other contexts) do not necessary measure the elapsed time (as in timed automata), but they may as well count some other data such as the distance covered by a car or the speed of a train. Transitions from a state to another depend also on the satisfaction of formulas by the actual counter values. Such a model is used in several papers, such as in [12]. For instance, Minsky machines [13] can be viewed as such multiple counter automata, which means that reachability is undecidable in general.

There are two ways to overcome this problem: either we restrict the class of models we consider or else we consider (lower or upper depending on the problem) approximations of the model. Essentially, these two points of view are not different: if we find some appropriate restriction of the models, then this corresponds to an appropriate class of approximations. The question then is to find a class which is as expressive as possible and for which reachability is still decidable. Assume for instance that counters may take integer values. Then we would like to describe sets of (reachable) configurations by Presburger formulas, assuming that the guards of the transitions are also expressed in Presburger arithmetic. This is not always possible because loops in the automaton yield fixed points which correspond to infinite disjunctions of Presburger formulas and actually, even with a single state, the set of reachable counters values can be a non-recursive set of integers (or reals if we consider real-valued counters).

In this paper, we consider a fragment (or an approximation) yielding a decidable class. Our main result is the following: assume that the counters values before (unprimed names) and after a transition (primed names) are solutions of conjunctions of atomic formulas of the form $x\#y' + c$ or $x\#y + c$ or $x'\#y' + c$ or $x\#c$ or $x'\#c$ where $c \in \mathbb{Z}$ (resp $c \in \mathbb{R}$) and $\# \in \{\leq, \geq, =, >, <\}$. Then we show that the fixed point of iterating a composition of such transitions is expressible in Presburger arithmetic. For automata with multiple nested iterations, then the same result holds provided that intermediate fixed points are expressible in the adequate fragment.

Related works

There are several authors who considered other fragments and other approximations. Let us briefly mention them and compare with our result. N. Halbwachs in [12] considers a similar model. A priori, the fixed point of a loop whose guard is $g(\mathbf{x}, \mathbf{x'})$ is the set of counters values which satisfy the infinite disjunction (n is the number of iterations) $\bigvee_{n=0}^{+\infty} (\exists \mathbf{x_1} \ldots \exists \mathbf{x_n}. g(\mathbf{x}, \mathbf{x_1}) \wedge \ldots g(\mathbf{x_n}, \mathbf{x'}))$. N. Halbwachs, following [7], considers a *widening* operation ∇ and he computes an upper approximation of the above infinite disjunction $\bigvee_{n=0}^{+\infty} P_i$ by considering the limit of $P_0, P_0 \nabla P_1, P_0 \nabla P_1 \nabla P_2 \ldots$ which is always reached after finitely many steps. Basically, the widening construction removes some of the constraints of either argument, until one of the constraint subsumes the other. Several strategies for computing the polyhedron are given in [12]. [1] Consider however the following example:

Example 1. There is only one state, one loop and two counters: initially the counters values satisfy $1 \leq x \leq 2, 1 \leq y \leq 2$. The guard of the transition is given by $x + 1 \leq x' \leq x + 2, y + 1 \leq y' \leq y + 2$. Successive values of the counters after each iteration are represented in figure 1. Following [12], we would get the

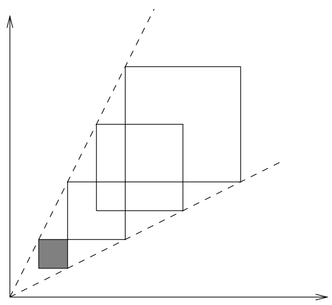

Fig. 1. Successive values of the counters

whole quarter of plan as an upper approximation. The exact computation is however possible, yielding $2y \geq x \wedge 2x \geq y$ (which is depicted on the figure using dashed lines). The guards satisfy our conditions, hence we will get this exact computation using our result.

P. Revesz in [15] also considers similar fixed point computation. The guards are of the form $x'_i \geq x_j + k$ with $k \geq 0$, which disallows for instance equalities: it is not possible to express e.g. $x' = x + 1$. On the other hand, he is able to handle several loops. The application of this result to verification is investigated in [11]. It is also extended, allowing periodicity conditions in [17].

B. Boigelot and P. Wolper consider in [4] guards of the form $x'_i = x_i + b$ plus additional guards involving only unprimed variables. They also get an exact fixed point computation in Presburger's arithmetic. Our result is more general in the sense that we may also have relations $x'_i \# x_j + b$ (i.e. relations between different counters) and inequalities as well. On the other hand, the precondition on unprimed variables is more general in [4] than in our result.

L. Fribourg and H. Olsen [10] consider a similar situation as in [4], except that their preconditions are less general. On the other hand, they consider the case of several loops, which is not the case in [4].

B. Boigelot in [3] characterizes precisely the functions from $\mathbb{Z}^n$ to $\mathbb{Z}^n$ of the form $f(x) = Ax + b$ such that the set of iterations of f is definable in weak monadic second order logic. This result is the most accurate one for the guards $x' = Ax + b$. However, there is no inequality here and no guard relating unprimed variables.

[1] even if the guards in [12] are only linear substitutions, it can be extended to certain kind of linear inequality, and for instance to our example.

In [5], the authors consider guards which can be arbitrary Presburger formulas. This model is more general than ours. However, they do not have any decidability result (the model is too expressive). They provide with approximations computations which yield semi-decision algorithms.

The paper is organized as follows. We start in section 2 with our model of multiple counters automata together with some examples of systems which are naturally expressed in this framework. We also explain the relationship with timed automata. Then we state our main result in section 3. Its proof is sketched in section 4. It relies on a careful analysis of shortest paths in a graph with an unbounded number of vertices. We show that shortest paths always lay in some particular sets of paths whose weights can be described by a Presburger formula. The full paper can be retrieved on
`http://www.lsv.ens-cachan.fr/~comon/ftp.articles/mca.ps`.

2 Multiple counters automata

In the following definition as well as in the rest of the paper, we consider integer valued counters. However, they can be real-valued as well without changing our results.

Definition 1. A *multiple counters automata* is a tuple $(Q, q_i, C, \delta \subseteq Q \times G(C, C') \times Q)$ where

- Q is a finite set of *states*
- $q_i \in Q$ is an *initial state*
- C is a finite set of *counter names*; C' is the set of primed counter names.
- $G(C, C')$ is the set of *guards* built on the alphabets C, C'. A member of $G(C, C')$ is a conjunction of atomic formulas of one of the forms $x \# y + c$, $x \# c$ where $x, y \in C \cup C'$, $\# \in \{\geq, \leq, =, >, <\}$. and $c \in \mathbb{Z}$. (or in $\mathbb{R}$)

A *configuration of the automaton* is a pair (q, v) where $q \in Q$ and v is a mapping from C into $\mathbb{N}$ (or $\mathbb{Z}$ or $\mathbb{R}$ or $\mathbb{R}_+$; as it is easy to see, this will not make any difference).

The automaton may *move* from a configuration (q, v) to a configuration (q', v'), which we write $(q, v) \to (q', v')$ if there is a triple $(q, g, q') \in \delta$ such that $v(C), v'(C') \models g$, with the standard interpretation of relational symbols.

Example 2. We consider a fragment[2] of the train example of [12]. On this example b is the number of beacons which have been encountered. It is given by the environment and measures the covered distance. s is the number of ticks which are sent by a global counter. Figure 2 shows transitions for which the train is on time an remains on time. g is the guard $b' = b, s' = s + 1, s \leq b + 8$ and h is the guard $b \leq s + 8, b' = b + 1, s' = s$. For instance, (1,(2,7)) and (1,(2,8)) are two possible consecutive configurations since $(2, 7), (2, 8) \models g$.

[2] This is because of size constraints of this paper. We consider the whole example,as well as some other examples in detail in the extended version of this paper.

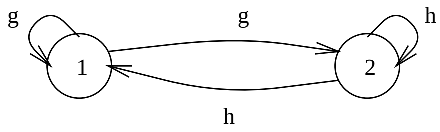

Fig. 2. A fragment of the train example

Safety analysis reduces to the computation of reachable configurations (or a superset of the reachable configurations). It is possible to compute this set, starting with inner loops and trying to compute meta-transitions. The concept of meta-transition, is presented in [4]. It amounts to consider a (possibly infinite) succession of elementary transitions of the automaton as a single transition, the guard being the conjunction of the guards of individual transitions (intermediate counter values being existentially quantified).

Example 3. Consider example 2. The sequence of transitions g^*g (from state 1 to state 2) can be replaced with a meta-transition g^+ whose guard is $s \leq b+8, b' = b, s' \leq b+9, s \leq s'-1$ and h^*h can be replaced with a meta-transition h^+ whose guard is $b \leq s+8, b \leq b'-1, b' \leq s+9, s' = s$.[3] g^+h^+ itself (which goes from 1 to itself) can be replaced with the meta transition whose guard is $s \leq s'-1, s' \leq b+9, b' \leq s'+9, b \leq b'-1$. Now, computing reachable configurations in state 1 reduces to compute the reachable configurations of a single state automaton with a single loop containing the computed guard.

Relationship with timed automata

At a first sight, timed automata are different from counter automata because the clocks always run at the same speed in the latter model whereas updates of the clocks seem to be possible only during a transition in the former model.

However, using a trick proposed recently by L. Fribourg [9], it is not difficult to translate timed automata into (real-valued) counter automata, at the price of adding a new clock, which is never reset. This translation does not change the structure of the automaton (transition and states are the same; the invariants and guards of the timed automata are used to compute the corresponding guard of the multiple counters automaton). Therefore, timed automata are a particular case of multiple clocks automata.

Further on, if we allow drifting clocks, then the simple above translation does not work any more, as it would yield guards of the form $\alpha \times x \leq y \leq \beta \times x$, which are not allowed in our model.

[3] The computation of meta-transitions can be performed using the result of the present paper.

3 Fixed point computations

Computing a meta transition for the composition of two transitions is an easy task. The main problem is to compute the fixed point of an iteration of a transition. Our main result, is that this is possible for a single loop, keeping the decidability of the computed guard:

Theorem 2. *Given a transition (q, g, q), there is an (effectively computable) Presburger arithmetic formula $\phi(C, C')$ such that $v, v' \models \phi(C, C')$ iff there exists an $n \in \mathbb{N}$ such that*

$$v, v' \models \exists C_1, \ldots, \exists C_n . g(C, C_1) \wedge \ldots \wedge g(C_n, C')$$

This result is not obvious as the formula $\exists n, \exists C_1, \ldots, \exists C_n . g(C, C_1) \wedge \ldots \wedge g(C_n, C')$ does not belong to Presburger arithmetic. It cannot be translated either (at least in an obvious way) into monadic second order logic: the counters vectors $C_1, \ldots, C_n$ are ordered (and their ordering is relevant), hence we cannot simply associate with each sequence of components a set of integers.

As usual in constraint solving we may represent inequalities $x \leq y + d$ using a graph whose vertices are the variables and edges are labeled with the delay d. (This is used for instance in many scheduling applications, see e.g. [6]). Here we have an unbounded number of variables: m variables for each of $C_1, \ldots, C_n$. m is known in advance. However, n is unbounded (and actually existentially quantified). Hence we consider a graph $G(g, n)$ whose number of vertices is unbounded $(n \times m)$. The purpose then is to compute a Presburger formula, which depends on n, and which expresses minimal paths in such a graph. As the number of vertices is unbounded, it is not possible to apply classical graph algorithms (such as Bellman-Ford [6]).

In the next section, which is devoted to the proof of this theorem, we develop a machinery to express these shortest paths; we first define the graph representation of the problem. Then we fold the unbounded graph into a finite (fixed) graph and investigate the relations between the graph and its folded version, showing to which extent paths in one graph are related with paths in the other graph.

Applications

Let us first state some consequences of the theorem. We say that a multiple counters automaton (or a part of an automaton) is *flat* if there is no nested loop in the transition graph.

Corollary 3. *Let $\mathcal{A}$ be a flat automaton and q, q' be two states of $\mathcal{A}$. Then there is a (effectively computable) Presburger formula $\phi_{q,q'}(\mathbf{x}, \mathbf{x'})$ such that $\mathbf{v}, \mathbf{v'} \models \phi_{q,q'}(\mathbf{x}, \mathbf{x'})$ iff $(q', \mathbf{v'})$ is accessible from $(q, \mathbf{v})$.*

In other words, the *binary* accessibility relation is definable in Presburger arithmetic, which yields, thanks to [8]:

Corollary 4. *The Model checking of* EF *formulas is decidable for (infinite) transitions systems that are defined by flat automata.*

The same results hold for real-valued counters. We only have to replace Presburger arithmetic with another theory; let $\mathcal{R}$ be the additive theory of real numbers with a predicate $\mathsf{Int}(x)$ which is satisfied by all integer values. This first-order theory is decidable, as it can be expressed in S1S (the monadic second-order logic), where real numbers are identified with infinite words (see e.g. [16]).

Corollary 5. *The binary accessibility relation for flat real-valued multiple counter automata is definable in* $\mathcal{R}$.

Then, for flat timed automata, the same result holds, thanks to Fribourg's trick [9].

4 Proof of theorem 2

4.1 Weighted graphs of unbounded size

First, for any guard, it is possible to assume without loss of generality that g is a conjunction of inequalities $x \leq y' + d$, $x' \leq y + d$, $x \leq y + d$, $x' \leq y' + d$. Indeed, strict inequalities can be replaced with non-strict inequalities, adding or removing 1 from the constant[4]. Equalities are replaced with two inequalities. Finally, we can take care of $x \leq c$, where c is a constant by adding a dummy counter whose value is always 0.

We consider weighted (directed) graphs $G(g, n)$ whose vertices are pairs (i, t) with $i \in [1..m]$ and $t \in [0..n]$ are integers, n being a parameter $n \geq 1$ and $m = |C|$. Given a guard g, the set of edges of $G(g, n)$ consists of the following pairs:

- for $i, j \in [1..m]$ and $t \in [0..n-1]$, $(i, t) \xrightarrow{d} (j, t+1)$ iff g contains an inequality $x_i \leq x'_j + d$.
- for $i, j \in [1..m]$ and $j \in [1..n]$, $(i, t) \xrightarrow{d} (j, t-1)$ iff g contains an inequality $x'_i \leq x_j + d$
- for $i, j \in [1..m]$ and $t \in [1..,n]$, $(i, t) \xrightarrow{d} (j, t)$ iff g contains an inequality $x'_i \leq x'_j + d$
- for $i, j \in [1..m]$ and $t \in [0..n-1]$, $(i, t) \xrightarrow{d} (j, t)$ iff g contains an inequality $x_i \leq x_j + d$.

Example 4. Consider again example 2 and the meta-transition of example 3. The graph corresponding to the new guard $s \leq s' - 1, s' \leq b + 9, b' \leq s' + 9, b \leq b' - 1$ is depicted on figure 3.

[4] For real-valued interpretations, strict inequalities cannot be removed and we have to consider in the graph both strict and non-strict inequalities. This complicates a little bit the picture, however nothing essential is changed.

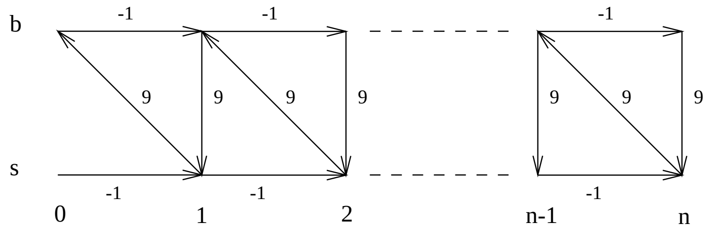

Fig. 3. The graph corresponding to a meta-transition of example 3

A *path* from a to b in a graph G is a finite sequence $e_1, \ldots, e_{N-1}$ of edges $a_i \xrightarrow{d_i} a_{i+1}$ such that $a_1 = a$ and $a_N = b$. A path is sometimes identified with the sequence of vertices $a_1, \ldots, a_N$ when there is no ambiguity.

Let $\psi_n(C, C')$ be the formula $\exists C_1, \ldots, \exists C_n . g(C, C_1) \wedge \ldots \wedge g(C_{n-1}, C')$. Then proving the theorem amounts to show how to compute a formula which is equivalent to $\exists n . \psi_n(C, C')$.

Let $\Gamma(i, t, i', t')$ be the set of all paths from (i, t) to (i', t') in $G(g, n)$. (This set can be infinite). The *weight* $w(\gamma)$ of a path γ is the sum of weights of all edges along the path.

The following lemma shows that we can eliminate intermediate steps, sticking to paths from a fixed number of vertices to a fixed number of vertices. However, the set of paths is still potentially infinite.

Lemma 6. $v, v' \models \psi_n(C, C')$ *iff for every indices* $i, j \in [1..m]$

- *for all paths* $\gamma \in \Gamma(i, 0, j, n)$, $v, v' \models x_i \leq x'_j + w(\gamma)$
- *for all paths* $\gamma \in \Gamma(i, n, j, 0)$, $v, v' \models x'_i \leq x_j + w(\gamma)$
- *for all paths* $\gamma \in \Gamma(i, 0, j, 0)$, $v, v' \models x_i \leq x_j + w(\gamma)$
- *for all paths* $\gamma \in \Gamma(i, n, j, n)$, $v, v' \models x'_i \leq x'_j + w(\gamma)$
- *for all paths* $\gamma \in \Gamma(i, n-1, i, n-1)$, $w(\gamma) \geq 0$, *if* $n \geq 2$.

4.2 The folded graph

In this section, we consider the "untimed" version of $G(g, n)$, which corresponds to identify vertices which only differ in their second components. Basically, we can compute on this "folded version" H of $G(g, n)$ a formula $\Phi_{i,j}$ which defines the set of all weights of paths from i to j.

Let H be the three-coloured weighted graph whose vertices are $[1..m]$ and whose edges are given by g:

- $i \overset{d}{\underset{1}{\rightarrow}} j$ if $x_i \leq x_j + d$ or $x'_i \leq x'_j + d$ is in g
- $i \overset{d}{\underset{2}{\rightarrow}} j$ if $x'_i \leq x_j + d$ is in g
- $i \overset{d}{\underset{3}{\rightarrow}} j$ if $x_i \leq x'_j + d$ is in g

For each edge e in G we associate in an obvious way an edge in H: $\pi((i,t) \xrightarrow{d} (j,t')) = i \xrightarrow[k]{d} j$ with $k=1$ if $t=t'$, $k=2$ if $t'<t$ and $k=3$ if $t'>t$. π is extended to paths of $G(g,n)$.

Example 5. Figure 4 shows the folded version of the graph given on figure 3.

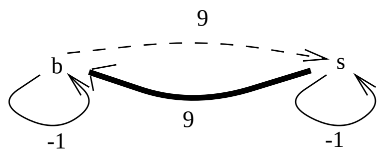

Fig. 4. The folded graph

Now, we need not to consider all paths in $\Gamma(i,t,j,t')$: according to lemma 6, we may only consider one path of H for each possible weight. Formally, if $\gamma \in \Gamma(i,t,j,t')$ and $\alpha(\gamma,e)$ is the multiplicity of e in $\pi(\gamma)$,

$$w(\gamma) = w(\pi(\gamma)) = \sum_{e \in H} \alpha(\gamma,e) \times w(e)$$

Conversely, if we give the multiplicities α_e of each edge e, there is a path corresponding to these multiplicities, iff they statisfy a given formula. Basically, such formula can be derived from Parikh's theorem [14]. That is what is stated in the next lemma. In what follows, $\boldsymbol{\alpha}$ (resp. $\mathbf{z}$) is a vector indexed by the set of edges of H: $\boldsymbol{\alpha} = (\alpha_{e_1}, ..., \alpha_{e_n})$.

Lemma 7. *For every i,j, there is a Presburger formula $\rho^{i,j}$ such that $\boldsymbol{\alpha} \models \rho^{i,j}$ iff there is a path γ from i to j in H such that $w(\gamma) = \sum_{e \in H} \alpha_e \times w(e)$.*

Remark: If H does not involve colour 1 and either does not involve colour 2 or colour 3, then we can already conclude the proof of theorem 2: from lemma 6, and since there is no (non-empty) path in $\Gamma(i,0,j,0) \cup \Gamma(i,n,j,n) \cup \Gamma(i,n-1,i,n-1)$ and either no path in $\Gamma(i,0,j,n)$ or no path in $\Gamma(i,n,j,0)$, we only have to find an equivalent Presburger formula for $\exists n. \bigwedge_{\gamma \in \Gamma(i,0,j,n)} x_i \leq x'_j + w(\gamma)$ (resp. $\exists n. \bigwedge_{\gamma \in \Gamma(i,n,j,0))} x'_i \leq x_j + w(\gamma)$). Such a formula would be

$$\exists n. \bigwedge_{i=1}^{m} \bigwedge_{j=1}^{m} \forall \mathbf{z}.(n = \sum_{e \in H} z_e \wedge \rho^{i,j}) \Rightarrow x_i \leq x'_j + \sum_{e \in H} z_e \times w(e)$$

However, in general, a path in H does not necessary correspond to a path in G. Relationships between paths of H and paths of $G(g,n)$ can be described more accurately. If γ is a path of H, we let $N_i(\gamma)$ be the number of edges of γ whose colour is i and $\delta_t(\gamma) = N_3(\gamma) - N_2(\gamma)$.

Lemma 8. *A path γ in H is the projection of some path in $G(g,n)$ only if there are $\delta_1, \delta_2 \in \mathbb{Z}$ such that $0 \leq \delta_2 - \delta_1 \leq n$ and for every prefix γ' of γ, $\delta_t(\gamma') \in [\delta_1..\delta_2]$.*

Conversely, if there are $\delta_1, \delta_2 \in \mathbb{Z}$ such that $0 \leq \delta_2 - \delta_1 \leq n$ and for every prefix γ' of γ, $\delta_t(\gamma') \in [\delta_1 + 1..\delta_2 - 1]$, then γ is the projection of a path in $G(g,n)$.

Now, the problem is that the property of prefixes given in lemma 8 cannot be characterized by counting the number of occurrences of each edge in a path only; we have to exploit the fact that we are only interested in minimal weight paths. Moreover, we will show that among the paths of minimal weight (relating two given vertices) there are "regular" ones, for which we can compute the boundedness condition of lemma 8.

4.3 Exploiting the quasi-ordering on paths

Now, we come to the hard part of the proof which cannot be detailed in this short paper. Let us only give the milestones. First, we rule out the case of cycles; if there is a cycle γ in H such that $w(\gamma) \geq 0$, it is not a minimal weighted path and if $w(\gamma) < 0$, the fixed point of the iteration is reached after a computable bounded number of steps :

Lemma 9. *If there is a cycle γ in H of length k such that $w(\gamma) < 0$ and $\delta_t(\gamma) = 0$, then $\psi_n(C, C')$ is unsatisfiable for $n \geq 1 + \frac{k}{2}$.*

Now, we give some transformation rules on paths, which preserve the weight of minimal weight paths, and whose termination is guaranteed by the cycle-freeness of G. One transformation consists in locally gathering together elementary cycles:

Lemma 10. $\forall \alpha, \exists B, \forall k \in [0..n - B], \forall i, j \in [0, m], \forall \gamma \in \Gamma(i, k, j, B + k), \exists \gamma' \in \Gamma(i, k, j, B + k), w(\gamma) = w(\gamma') \wedge \pi(\gamma') = c_1.(\sigma)^{\alpha}.c_2$ *with σ an elementary cycle of H.*

Next, we apply two rules which duplicate or remove elementary cycles, yielding paths of smaller (or equal) weight and which are more "regular" (see fig 5).

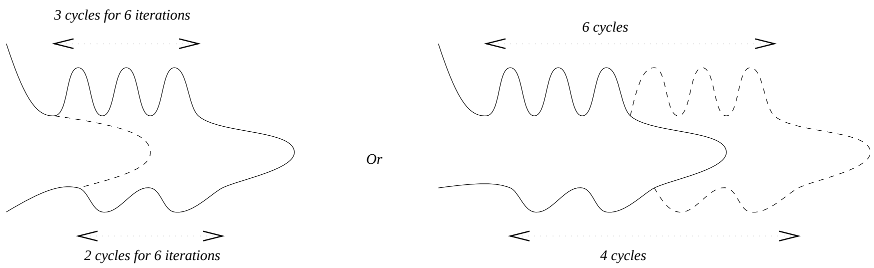

Fig. 5. how to construct the regular path

If the path is long enough, then we get a normal form which is depicted on figure 6 : the paths η_k and η'_k have a width bounded by a constant B (independent from n) and θ_k, θ'_k go from one extremity to the other. They consist themselves of a path of length bounded by B followed by an iteration of a particular cycle of H, followed by a bounded length path. Moreover, the number of such paths η_k is smaller than the number of counters.

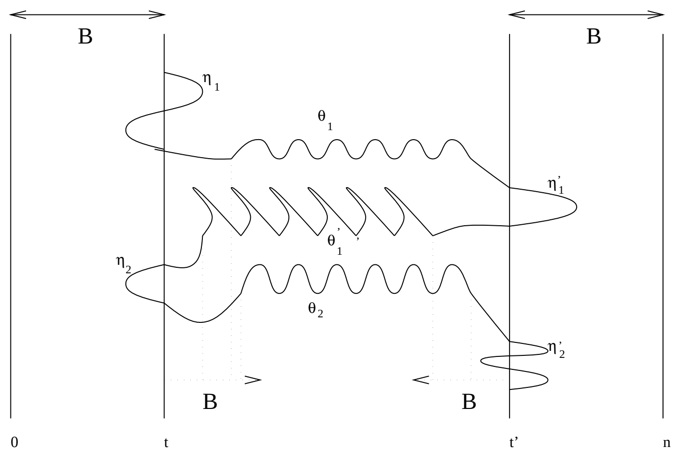

Fig. 6. Only paths going "back and forth" have to be considered

Then when n is large enough ($n \geq 2B$), $\psi_n(C, C')$ is logically equivalent to a formula

$$\exists \mathbf{x^B}, \exists \mathbf{x^{n-B}}, \Phi_1(\mathbf{x}, \mathbf{x^B}) \wedge \Phi_2(\mathbf{x^B}) \wedge \Phi_3(\mathbf{x^B}, \mathbf{x^{n-B}}) \wedge \Phi_2(\mathbf{x^{n-B}}) \wedge \Phi_1(\mathbf{x^{n-B}}, \mathbf{x'})$$

Φ_1 expresses the constraints generated by B iterations of the loop Φ_2 expresses the constraints between counters values after B iterations and which correspond in the graph to paths η, whose width is bounded by K. Φ_3 expresses paths that are the iteration of elementary cycles of H, and which brings from the $(B+i)$ths values of the counters to the $n-B-j$ths values of the counters, for $i, j \in [0, B]$.

5 Conclusion

We have shown that the fixed point of a single loop of multiple counters automata is definable in the additive theory of $\mathbb{N}$ (resp. $\mathbb{R}$, resp. $\mathbb{Z}$). Thanks to this result, it is possible to compute the exact set of reachable configurations in a number of situations for which such a computation was unknown. This also provides better approximations for the general case. The complexity of the resulting reachability analysis is high (a double exponential in the number of counters since the length of the formula is exponential in the number of counters in the worst case). It seems however to be manageable on the examples: the upper bound needs not to be met on all examples.

Acknowledgements

We acknowledge L. Fribourg for many discussions on multiple counters automata.

References

1. R. Alur and D. Dill. Automata for modeling real-time systems. In *Proc. 17th Int. Coll. on Automata, Languages and Programming, Warwick, LNCS 443*, pages 322–335. Springer-Verlag, 1990.
2. O. Bernholtz, M. Vardi, and P. Wolper. An automata-theoretic approach to branching time model checking. In *Proc. Computer Aided Verification*, 1994.
3. B. Boigelot. Linear operators and regular languages (ii). Unpublished draft, jan 1997.
4. B. Boigelot and P. Wolper. Symbolic verification with periodic sets. In *Computer Aided Verification, Proc. 6th Int. Conerence*, LNCS, Stanford, June 1994. Springer-Verlag.
5. T. Bultan, R. Gerber, , and W. Pugh. Symbolic model checking of infinite state systems using presburger arithmetic. In O. Grumberg, editor, *Proc. Computer Aided Verification*, volume 1254 of *LNCS*, Haifa, Israel, 1997. Springer-Verlag.
6. T. Cormen, C. Leiserson, and R. Rivest. *Introduction to algorithms.* MIT Press, 1990.
7. P. Cousot and N. Halbwachs. Automatic discovery of linear restraints among variables of a program. In *Proc. Int. Conf. on Princinples Of Programming Languages (POPL)*, 1978.
8. J. Esparza. Decidability of model checking for infinite-state concurrent systems. *Acta Informatica*, 34:85–107, 1997.
9. L. Fribourg. A closed form evaluation for extending timed automata. Technical Report 1998-02, Laboratoire Spécification et Vérification, ENS Cachan, Mar. 1998.
10. L. Fribourg and H. Olsen. A decompositional approach for computing least fixed-point of datalog programs with z-counters. *J. Constraints*, 1997.
11. L. Fribourg and J. Richardson. Symbolic verification with gap-order constraints. Research Report LIENS-96-3, Ecole Normale Supérieure, Paris, Feb. 1996.
12. N. Halbwachs. Delay analysis in synchronous programs. In *Proc. Computer Aided Verification*, LNCS 697, pages 333–346. Springer-Verlag, 1993.
13. M. Minsky. *Computation, Finite and Infinite Machines.* Prentice Hall, 1967.
14. R. Parikh. On context-free languages. *J. ACM*, 13, 1966.
15. P. Revesz. A closed form for datalog queries with integer order. In *Proc 3rd International Conference on Database Theory*, pages 187–201, Paris, 1990.
16. W. Thomas. Automata on infinite objects. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, pages 134–191. Elsevier, 1990.
17. D. Toman, J. Chomicki, and D. S. Rogers. Datalog with integer periodicity constraints. In *Int. Symp. on Logic Programming*, 1994.