Reachability in Succinct and Parametric One-Counter Automata

Christoph Haase, Stephan Kreutzer, Joël Ouaknine, and James Worrell

Oxford University Computing Laboratory, UK
{chrh,kreutzer,joel,jbw}@comlab.ox.ac.uk

Abstract. One-counter automata are a fundamental and widely-studied class of infinite-state systems. In this paper we consider one-counter automata with counter updates encoded in binary—which we refer to as the succinct encoding. It is easily seen that the reachability problem for this class of machines is in PSPACE and is NP-hard. One of the main results of this paper is to show that this problem is in fact in NP, and is thus NP-complete.
We also consider parametric one-counter automata, in which counter updates be integer-valued parameters. The reachability problem asks whether there are values for the parameters such that a final state can be reached from an initial state. Our second main result shows decidability of the reachability problem for parametric one-counter automata by reduction to existential Presburger arithmetic with divisibility.

1 Introduction

Counter automata are a fundamental computational model, known to be equivalent to Turing machines [19], and there has been considerable interest in subclasses of counter machines for which reachability is decidable, such as Petri nets, one-counter automata and flat counter automata [5, 18]. As originally conceived by Minsky, counters are updated either by incrementation or decrementation instructions. However, for many applications of counter machines, including modelling computer programs, it is natural to consider more general types of updates, such as adding integer constants to a counter [3, 5, 16] or adding integer parameters [4, 12]. Parametric automata are used in various synthesis problems, and to model open programs, whose behaviour depends on values input from the environment [2]. In [20] parameters are also used to model resources (e.g., time, memory, dollars) consumed by transitions. The reachability problem for parametric counter automata asks whether there exist values of the parameters such that a given configuration is reachable from another given configuration.

In this paper we show NP-completeness of the reachability problem for one-counter automata in which counters can be updated by adding integer constants, where the latter are encoded in binary. We also show decidability of reachability for parametric one-counter automata by reduction to existential Presburger arithmetic with divisibility [17]. We defer consideration of the complexity of the latter problem to the full version of this paper.

1.1 Related Work

The verification literature contains a large body of work on decidability and complexity for various problems on restricted classes of counter automata. The work that is closest to our own is that of Demri and Gascon on model checking extensions of LTL over one-counter automata [8]. They consider automata with one integer-valued counter, with updates encoded in unary, and with sign tests on the counter. They show that reachability in this model is NL-complete. Determining the complexity of reachability when updates are encoded in binary is posed as an open problem by Demri in [7], Page 61, Problem 13. Since this last problem assumes an integer-valued counter with sign tests, it is more general than the one considered in our Theorem 1, and it remains open.

Another work closely related to our own is that of Ibarra, Jiang, Tran and Wang [12], which shows decidability of reachability for a subset of the class of deterministic parametric one-counter automata with sign tests. The decidability of reachability over the whole class of such automata is stated as an open problem in [12]. Note that although we do not allow negative counter values and sign tests, we allow nondeterminism. Thus our Theorem 4 is incomparable with this open problem.

Aside from reachability, similarity and bisimilarity for one-counter automata and one-counter nets have been considered in [1, 13, 14], among others.

For automata with more than one counter, other restrictions are required to recover decidability of the reachability problem: for example, *flatness* [5, 16] and *reversal boundedness* [11]. Bozga, Iosif and Lakhnech [4] show decidability of the reachability problem for flat parametric counter automata with a single loop, by reduction to a decidable problem concerning quadratic diophantine equations. Such systems of equations also feature in the work of Ibarra and Dang [11]. They exhibit a connection between a decidable class of quadratic diophantine equations and a class of counter automata with reversal-bounded counters.

2 One-Counter Automata

A **one-counter automaton** is a nondeterministic finite-state automaton acting on a single counter which takes values in the nonnegative integers. Formally a one-counter automaton is a tuple $\mathcal{C} = (V, E, \lambda)$, where V is a finite set of control locations, $E \subseteq V \times V$ is a finite set of transitions, and $\lambda : E \to Op$ is a function that assigns to each transition an operation from the set $Op = \{\mathsf{zero}\} \cup \{\mathsf{add}(a) : a \in \mathbb{Z}\}$. The operation zero represents a zero test on the counter, whereas $\mathsf{add}(a)$ denotes the operation of adding a to the value of the counter.

A **configuration** of the counter automaton $\mathcal{C}$ is a pair (v, c), where $v \in V$ is a control location and $c \in \mathbb{N}$ is the value of the counter. The transition relation on the locations of $\mathcal{C}$ induces an unlabelled transition relation on configurations in the obvious way: an edge $(v, v') \in E$ with labelled zero yields a single transition $(v, 0) \longrightarrow (v', 0)$, while the same edge with label $\mathsf{add}(a)$ yields a transition $(v, c) \longrightarrow (v', c + a)$, provided that both c and $c + a$ are both non-negative.

A **computation** π of a counter automaton $\mathcal{C}$ is a finite sequence of transitions between configurations

$$\pi = (v_0, c_0) \longrightarrow (v_1, c_1) \longrightarrow \cdots \longrightarrow (v_n, c_n)\,.$$

We define the length of π to be $\text{length}(\pi) = n$. We sometimes write $\pi : (v_0, c_0) \longrightarrow^* (v_n, c_n)$ or $(v_0, c_0) \xrightarrow{\pi} (v_n, c_n)$ to denote that π is a computation from (v_0, c_0) to (v_n, c_n).

The **reachability problem** asks, given a one-counter automaton $\mathcal{C}$ and configurations (v, c) and (v', c'), whether there is a computation starting in (v, c) and ending in (v', c'). The **control-state reachability problem** asks, given $\mathcal{C}$ and two locations v and v', whether there is a computation from $(v, 0)$ to (v', c') for some counter value c'. It is easily seen that both reachability problems are reducible to each other in logarithmic space.

In determining the complexity of these problems we assume a standard encoding of counter automata and their configurations—in particular, we suppose that integers are encoded in binary. Given a counter machine $\mathcal{C}$, we denote by $|\mathcal{C}|$ the length of the encoding of $\mathcal{C}$. It is easy to see that the shortest computation between two given locations may have length exponential in $|\mathcal{C}|$. For example, in the automaton in Figure 2 the unique path from $(v_0, 0)$ to $(v_1, 0)$ has length 2^n.

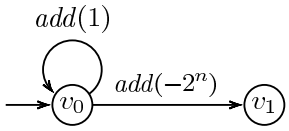

Fig. 1.

The first observation is the following.

Proposition 1. *The reachability problem for one-counter automata is* NP*-hard.*

Proof. The proof is by reduction from the SUBSET SUM problem [9]. Recall that an instance of the latter consists of a set of positive integers $S = \{a_1, a_2, \ldots, a_n\}$ and a goal c, and the question asked is whether there exists a subset $T \subseteq S$ such that $\sum T = c$. This reduces to the question of whether configuration (v_n, c) is reachable from $(v_0, 0)$ in the one-counter automaton in Figure 2. □

Proposition 1 crucially depends on encoding integers in binary. Indeed, it follows from Proposition 2, below, that if integers are encoded in unary then the reachability problem becomes NL-complete, since it reduces to reachability in a polynomial-size graph.

The first main contribution of this paper is to establish an upper bound for the complexity of the reachability problem, matching the lower bound in Proposition 1.

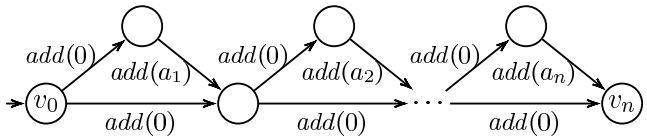

Fig. 2. Reduction from SUBSET SUM to reachability.

Theorem 1. *The reachability problem for one-counter automata is in* NP.

The idea behind the proof of Theorem 1 is as follows. Suppose one is given a one-counter automaton $\mathcal{C}$, and two configurations (v, c) and (v', c'). One can show that if (v', c') is reachable from (v, c), then there is a computation π from (v, c) to (v', c') whose length is bounded by an exponential function in $|\mathcal{C}|$ and the bit lengths of c and c'. This computation has a succinct certificate: a *network flow* which records for each edge of $\mathcal{C}$ how many times it is taken in π. This flow has a polynomial-size description, and so it can be guessed in polynomial time. The main difficulty in fleshing out this idea is the problem of how to validate such a certificate; that is, given a flow, to determine in polynomial time whether it arises from a valid computation of the counter machine. To solve this problem we define a subclass of such flows with certain structural properties, called *reachability certificates*. We show that validating reachability certificates can be done in NP, and that the computation π, above, can, without loss of generality, be divided into sub-computations, each of which generates a reachability certificate.

2.1 Parametric Counter Automata

A **parametric one-counter automaton** is a tuple $\mathcal{C} = (V, E, X, \lambda)$, where the sets V and E of vertices and edges are as in the definition of a one-counter automaton, X is a set of non-negative integer parameters, and the labelling function $\lambda : E \to Op$ has codomain

$$Op = \{\mathsf{zero}\} \cup \{\mathsf{add}(a), \mathsf{add}(x), \mathsf{add}(-x) : a \in \mathbb{Z}, x \in X\} .$$

The only difference with one-counter automata is the ability to add or subtract the value of a parameter $x \in X$ to the counter. Each instantiation of the parameters yields a different one-counter automaton.

The **reachability problem for parametric one-counter automata** asks, given configurations (v, c) and (v', c'), whether there exist values for the parameters such that there is a computation from (v, c) to (v', c'). We exhibit reductions in both directions between this problem and the satisfiability problem for the existential fragment of **Presburger arithmetic with divisibility**, i.e., the existential theory of the structure $(\mathbb{Z}, <, |, +, 0, 1)$, where $|$ is the binary *divides* predicate. Lipshitz [17] gave a procedure for deciding satisfiability of this logic. Thus we obtain our second main result.

Theorem 2. *The reachability problem for parametric one-counter automata is decidable.*

The reduction from existential Presburger arithmetic with divisibility to the reachability problem for parametric one-counter automata is fairly straightforward, and is detailed below. It follows a similar pattern to [2], which reduces existential Presburger arithmetic with divisibility to the reachability problem for two-clock parametric timed automata.

Let φ be a quantifier-free formula of Presburger arithmetic with divisibility. Without loss of generality, assume that φ is a positive Boolean combination of atomic formulas, $A = B$, $A < B$, $A \mid B$ and $\neg(A \mid B)$, where A and B are linear expressions in variables $x_1, \ldots, x_n$. By representing arbitrary integers as differences of positive integers we can also assume that the variables $x_1, \ldots, x_n$ range over the positive integers.

For each such atomic sub-formula ψ we construct a one-counter automaton $\mathcal{C}_\psi$, with parameters $x_1, \ldots, x_n$ and distinguished locations u and v, such that $(v, 0)$ is reachable from $(u, 0)$ iff ψ is satisfied. We can then combine the automata representing atomic sub-formulas using sequential composition to model conjunction and nondeterminism to model disjunction.

For an atomic formula $\psi \equiv A \mid B$, the automaton $\mathcal{C}_\psi$ first guesses the sign of A and B. Assume that A and B are guessed to be non-negative; the remaining cases are similar. In this case the automaton simply loads B into its counter and repeatedly subtracts A until the counter reaches 0.

For an atomic formula $\psi \equiv \neg(A \mid B)$ the automaton $\mathcal{C}_\psi$ first guesses the sign of A and B. Again, assume that A and B are non-negative. Then the automaton loads B into its counter and repeatedly subtracts A until the counter reaches a value strictly between 0 and A. It can be checked whether the counter is strictly between 0 and A by performing the following sequence of transitions: subtract one; add two; add one a nondeterministic number of times; subtract A; test for zero.

Handling the other atomic formulas is equally straightforward.

3 Weighted Graphs and Flow Networks

In this section we recall some standard definitions about weighted graphs and flow networks.

A **weighted graph** is a tuple $G = (V, E, w)$, where V is a set of vertices, $E \subseteq V \times V$ is a set of directed edges, and $w : E \to \mathbb{Z}$ assigns an integer **weight** to each edge. Given such a graph and two distinguished vertices $s, t \in V$, a **path** π from s to t, also called an s-t path, is a sequence of vertices $\pi = v_0 v_1 \ldots v_n$ with $v_0 = s$, $v_n = t$ and $(v_i, v_{i+1}) \in E$ for $0 \leq i < n$. A path with the same first and last vertices is called a **cycle**. To indicate that π is an s-t path we often write $\pi : s \longrightarrow^* t$. If $\pi : s \longrightarrow^* t$ and $\pi' : t \longrightarrow^* u$, then $\pi \cdot \pi'$ denotes the path from s to u obtained by composing π and π'. Given a cycle ℓ on vertex v, we define $\ell^0 = v$ (the trivial cycle on v) and $\ell^{n+1} = \ell^n \cdot \ell$ for $n \in \mathbb{N}$.

The **weight** of a path π, denoted weight(π) is the sum of the weights of the edges in π. If ℓ is a cycle such that weight(ℓ)>0 then we say that ℓ is a **positive cycle**, and if weight(ℓ)<0 then we say that ℓ is a **negative cycle**.

Given a weighted graph $G=(V,E,w)$, with distinguished vertices s and t, a **flow** from s to t, also called an s-t flow, is a function $f : E \to \mathbb{N}$ satisfying the following flow conservation condition for each vertex $u \in V - \{s,t\}$:

$$\sum_{(v,u)\in E} f(v,u) = \sum_{(u,v)\in E} f(u,v)\,.$$

The **value** $|f|$ of flow f is the net flow out of the source s (equivalently the net flow into the sink t), that is,

$$|f| = \sum_{(s,u)\in E} f(s,u) - \sum_{(u,s)\in E} f(u,s)\,.$$

The **weight** of the flow f is defined to be

$$\text{weight}(f) = \sum_{e\in E} f(e)\cdot w(e)\,.$$

An s-t path π determines an s-t flow f_π, where for each edge $e \in E$, $f(e)$ is defined to be the number of times edge e is taken in π. We call the class of flows that arise in this way **path flows.** Just as paths can be sequentially composed, path flows can be composed by summation: given an s-t path flow f and a t-u path flow g, we define a s-u path flow $f+g$ by $(f+g)(e) = f(e)+g(e)$ for each edge $e \in E$.

The **skew transpose** G^{op} of G is the weighted graph obtained by multiplying all edge weights by -1 and then reversing the direction of each edge. A path flow f from s to t in graph G induces a path flow f^{op} from t to s in G^{op}, where $f^{op}(u,v) = f(v,u)$.

4 Reachability Certificates

The following result [15, Lemma 42] shows that the reachability problem for one-counter machines is in PSPACE.

Proposition 2. *There is a polynomial P such that given a one-counter automaton $\mathcal{C}$ and configurations (v,c) and (v',c'), if (v,c) can reach (v',c') then there is computation from (v,c) to (v',c') of length at most $2^{P(n)}$, where n is the maximum of $|\mathcal{C}|$ and the bit lengths of c and c'.*

Let $\mathcal{C}=(V,E,\lambda)$ be a one-counter automaton. For proving NP-membership of the reachability problem, it is no loss of generality to assume that $\mathcal{C}$ has no zero tests. Indeed, since we may assume that each zero test is taken at most once, by guessing the order in which the zero tests are taken, a reachability query on a one-counter automaton with zero tests can be reduced to a linear number of

reachability queries on the same automaton with zero tests erased. Now a one-counter automaton without zero tests is nothing but a weighted graph, where the weight of an edge labelled $\mathsf{add}(a)$ is $a \in \mathbb{Z}$. For emphasis, we denote automaton $\mathcal{C}$ *qua* weighted graph by $G_{\mathcal{C}}$.

Recall that a computation π of $\mathcal{C}$ determines a path flow f_π in $G_{\mathcal{C}}$, mapping each edge to its multiplicity in π. If the length of π is bounded by an exponential function in the size of $\mathcal{C}$, then f_π has a description that is polynomial in the size of $\mathcal{C}$. We regard f_π as a polynomial *reachability certificate*. In this section we consider the problem of how to validate such a certificate in polynomial time; that is, given configurations (v, c) and (v', c'), we seek necessary and sufficient conditions on a flow f for there to exist a computation π from (v, c) to (v', c') with $f = f_\pi$, and we require that these conditions be polynomial-time checkable.

As a starting point, we recall the following straightforward variant of Euler's theorem.

Proposition 3. *Given vertices $s \neq t$, an s-t flow f is a path flow if and only if $|f| = 1$ and the subgraph induced by the set of edges $\{e \in E : f(e) > 0\} \cup \{(t, s)\}$ is strongly connected.*

Proposition 3 gives a way to check in linear time, given a flow f, whether there exists a path π such that $f = f_\pi$. The difficult part is then to determine whether π can be chosen such that it corresponds to a computation between given source and target configurations (v, c) and (v', c'). Informally speaking, we need to know that taking π from (v, c) does not cause the counter to go negative. More formally, given a path $\pi = v_0 v_1 \dots v_n$, define vertex v_j to be a **minimum** of π if the path $\pi' = v_0 v_1 \dots v_j$ has minimal weight among all prefixes of π; in this case we define $\mathrm{drop}(\pi)$ to be $\mathrm{weight}(\pi')$. Then π corresponds to a computation from (v, c) to (v', c') if and only if $\mathrm{drop}(\pi) \geq -c$ and $\mathrm{weight}(\pi) = c' - c$.

Given a path π from v to v', if there is a computation over π starting in configuration (v, c) and ending in configuration (v', c'), we say that π can be **taken from** (v, c) and **taken to** (v', c'). Next we introduce two key notions about flows which will help us to state sufficient conditions for a flow to be realisable by a computation between given configurations.

Given a flow f in $G_{\mathcal{C}}$, a **cycle** in f is a cycle ℓ in $G_{\mathcal{C}}$ such that f assigns positive flow to each edge in ℓ. If ℓ has positive (resp. negative) weight, then we speak of f having a positive (resp. negative) cycle.

Let f be a path flow from s to t. A **decomposition** of f consists of an enumeration $v_1, \dots, v_n$ of the set set $\{v : \exists u. f(u, v) > 0\}$ of vertices with incoming flow, together with a sequence of flows $f_0, \dots, f_{n-1}$ such that (i) f_0 is a path flow from s to v_1, (ii) f_i is a path flow from v_i to v_{i+1} for $1 \leq i \leq n-1$, (iii) $f = f_0 + f_1 + \dots + f_{n-1}$, and (iv) if $i \leq j$ then f_j directs no flow into vertex v_i.

Proposition 4. *Let (v, c) and (v', c') be configurations of $\mathcal{C}$ and f be a path flow in $G_{\mathcal{C}}$ from v to v' such that* $\mathrm{weight}(f) = c' - c$.

(i) *If f has no positive cycles, then $f = f_\pi$ for some computation $\pi : (v, c) \longrightarrow^* (v', c')$ if and only if there is a decomposition $f = f_0 + \ldots + f_{n-1}$ such that $\sum_{i=0}^{j} \text{weight}(f_i) \geq -c$, $0 \leq j < n$.*

(ii) *If f has no negative cycles, then $f = f_\pi$ for some computation $\pi : (v, c) \longrightarrow^* (v', c')$ if and only if there is a decomposition $f^{op} = f_0 + \cdots + f_{n-1}$ in $G_{\mathcal{C}}^{op}$ such that $\sum_{i=0}^{j} \text{weight}(f_i) \geq -c'$, $0 \leq j < n$.*

Proof (sketch).

(i) Since f has no positive cycles, any path π in $G_{\mathcal{C}}$ such that $f = f_\pi$ also has no positive cycles. Thus in a computation along π, the net change in the counter value between consecutive visits to a given location is less than or equal to 0. Thus to check that the counter never becomes negative, we need only verify that it is non-negative the last time π visits any given location. It is not hard to see that there exists a path π satisfying this last condition if and only if f has a flow decomposition satisfying the condition in *(i)* above.

(ii) This follows by applying the result stated in Part *(i)* to the flow f^{op} on the skew transpose of $G_{\mathcal{C}}$.

□

In a slightly different vein to Proposition 4, Proposition 5 gives a simple condition on $G_{\mathcal{C}}$, rather than on the flow f, that guarantees that (v', c') is reachable from (v, c).

Proposition 5. *Let (v, c) and (v', c') be configurations of $\mathcal{C}$ and f be a path flow in $G_{\mathcal{C}}$ from v to v' such that* $\text{weight}(f) = c' - c$. *If there is a positive cycle ℓ that can be taken from (v, c), and a negative cycle ℓ' that can be taken to (v', c'), then (v', c') is reachable from (v, c).*

Proof (sketch). The idea is simple. By definition, there exists a path π from v to v' in $G_{\mathcal{C}}$ such that $f = f_\pi$. Now π need not yield a computation from (v, c) to (v', c') since it may be that $\text{drop}(\pi) \leq -c$. However we can circumvent this problem, and build a computation from (v, c) to (v', c'), by first *pumping up* the value of the counter by taking the positive cycle ℓ a number of times, then traversing π, and then *pumping down* the value of the counter by taking the negative cycle ℓ' a number of times. Note that if we take the positive cycle $-k \cdot \text{weight}(\ell')$ times, and the negative cycle $k \cdot \text{weight}(\ell)$ times, for some positive integer k, then the net effect on the counter is 0. □

A flow f is called a **reachability certificate** for two configurations (v, c) and (v', c') if there exists a path $\pi : (v, c) \longrightarrow^* (v', c')$ such that $f = f_\pi$ and one of the following three conditions holds: (i) f has no positive cycles; (ii) f has no negative cycles; (iii) there exists a positive cycle ℓ that can be taken *from* (v, c) and a negative cycle ℓ' that can be taken *to* (v', c'). Depending on which of the above three cases holds, we respectively call f_π a type-1, type-2 or type-3 reachability certificate. In any case, we say that the computation π *yields* the reachability certificate f_π. The following corollary of Propositions 4 and 5 gives an upper bound on the complexity of recognising a reachability certificate.

Corollary 1. *Given a one-counter machine $\mathcal{C}$, two configurations (v,c) and (v',c'), and a path flow f in $G_{\mathcal{C}}$, the problem of deciding whether f is a reachability certificate for (v,c) and (v',c') is in* NP.

Proof. It can be checked in polynomial time whether f has any positive cycles or any negative cycles, e.g., using the Bellman-Ford algorithm [6]. If f has no positive cycles, then by Proposition 4(i) to show that f is a type-1 reachability certificate we need only guess a decomposition $f = f_0 + \cdots + f_{n-1}$ such that $\sum_{i=0}^{j} \text{weight}(f_i) \geq -c$, $0 \leq j < n$. The case that f has no negative cycles similarly uses Proposition 4(ii).

It remains to consider type-3 reachability certificates. To this end, observe that there is a positive cycle ℓ that can be taken from (v,c) if and only if there is a positive simple cycle in the same strongly connected component of $G_{\mathcal{C}}$ as v that can be reached and taken from (v,c). This last condition can be checked in polynomial time using a small modification of the Bellman-Ford algorithm. By running the same algorithm on the skew transpose of $G_{\mathcal{C}}$, it can be checked whether there is a negative cycle ℓ' that can be taken to (v',c'). □

Note that we do not assert that the existence of a computation $\pi : (v,c) \longrightarrow^* (v',c')$ guarantees that there is a reachability certificate for (v,c) and (v',c'). However, in the next section we show that the existence of a computation from (v,c) to (v',c') can be witnessed using at most three polynomial-size reachability certificates.

5 NP-Membership

Based on the ideas developed in the previous section, we are interested in paths π for which the associated flow f_π has no positive cycles or no negative cycles. It is important to note here that f_π may have positive cycles even though π itself does not have any positive cycles (and similarly for negative cycles). We will use the following proposition to overcome this problem.

Proposition 6. *Let π be a computation from (v,c) to (v',c') in which all cycles are negative. Then either the corresponding flow f_π has no positive cycles, or there is a computation $\theta = \theta_1 \cdot \theta_2 \cdot \theta_3$ from (v,c) to (v',c') such that* $\text{length}(\theta_1) < \text{length}(\pi)$ *and θ_2 is a positive cycle.*

Proof. Suppose that f_π contains a positive cycle ℓ. Let $u \in V$ be the first vertex of ℓ that π reaches, and let the counter value be y when π first reaches u. We claim that there is a positive cycle in $G_{\mathcal{C}}$ that can be taken from configuration (u,y).

If ℓ cannot be taken from (u,y) then we argue as follows. Factor ℓ as $\ell = u \xrightarrow{\rho_1} w \xrightarrow{\rho_2} u$, with w a minimum of ℓ (cf. Figure 5, which depicts the height of the counter as ℓ is traversed). Then we have $\text{weight}(\rho_1) < -y$. But π must visit w after it first visits u (since u is the first vertex of ℓ visited by π), so there is a

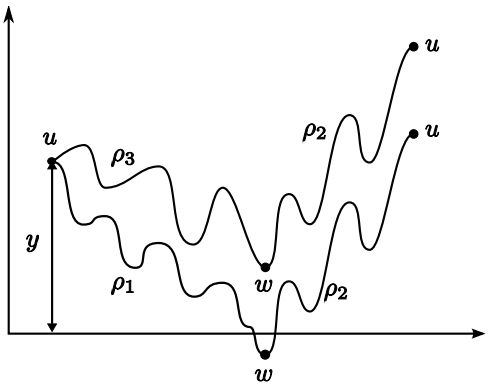

Fig. 3. Decomposition of the loop ℓ.

path $\rho_3 : u \longrightarrow^* w$ in $G_{\mathcal{C}}$ such that $\text{weight}(\rho_3) \geq \text{drop}(\rho_3) \geq -y > \text{weight}(\rho_1)$. Now consider the cycle $\ell' : u \xrightarrow{\rho_3} w \xrightarrow{\rho_2} u$. The preceding inequality gives

$$\begin{aligned} \text{weight}(\ell') &= \text{weight}(\rho_3) + \text{weight}(\rho_2) \\ &> \text{weight}(\rho_1) + \text{weight}(\rho_2) \\ &= \text{weight}(\ell)\,, \end{aligned}$$

so that ℓ' is a positive cycle. We also have

$$\begin{aligned} \text{drop}(\ell') &\geq \text{drop}(\rho_3) + \text{drop}(\rho_2) \\ &\geq -y + 0 \\ &= -y\,, \end{aligned}$$

whence ℓ' can be taken from (u, y). This proves the claim.

Next we observe that the first occurrence of u in π actually lies on a negative cycle in π. This is because π must visit u again, and all cycles in π are negative by assumption. Thus we can factor π as

$$(v, c) \xrightarrow{\pi_1} (u, y) \xrightarrow{\pi_2} (u, y') \xrightarrow{\pi_3} (v', c')$$

such that there is a positive cycle ℓ' that can be taken from (u, y), and with π_2 a negative cycle.

To define the required computation $\theta = \theta_1 \cdot \theta_2 \cdot \theta_3$, we reuse an idea from the proof of Proposition 5. Write $\text{weight}(\ell') = p$ and $\text{weight}(\pi_2) = -q$, where $p, q > 0$. Then define $\theta_1 = \pi_1$, $\theta_2 = (\ell')^q$ and $\theta_3 = (\pi_2)^{p+1} \cdot \pi_3$. Clearly $\text{length}(\theta_1) < \text{length}(\pi)$ and θ_2 is a positive cycle, as required. Since the positive cycle $(\ell')^q$ is cancelled out by the negative cycle $(\pi_2)^p$, θ is a computation from (v, c) to (v', c'). □

We also have the following dual of Proposition 6.

Proposition 7. *Let π be a computation from (v,c) to (v',c') in which all cycles are positive. Then either the corresponding flow f_π has no negative cycles, or there is a computation $\theta = \theta_1 \cdot \theta_2 \cdot \theta_3$ from (v,c) to (v',c') such that θ_2 is a negative cycle and* $\text{length}(\theta_3) < \text{length}(\pi)$.

Next we exploit Propositions 6 and 7 to show that the reachability of a configuration (v',c') from a configuration (v,c) can be witnessed by at most three reachability certificates.

Proposition 8. *If (v',c') is reachable from (v,c), then there exists a computation π from (v,c) to (v',c') that can be written $\pi = \pi_1 \cdot \pi_2 \cdot \pi_3$, such that π_1, π_2 and π_3 each yield reachability certificates.*

Proof. Let $\pi = v_0 v_1 \dots v_n$ be (the path underlying) a computation from (v,c) to (v',c'). Without loss of generality we assume that π contains no zero-weight cycles. If π contains a positive cycle, then define i_1 such that v_{i_1} is the first vertex that appears in a positive cycle in π; otherwise let $i_1 = n$. Write $\pi_1 = v_0, v_1, \dots, v_{i_1}$ and assume that π is chosen such that $\text{length}(\pi_1)$ is minimised. Then π_1 contains only negative cycles; thus from Proposition 6 and the minimality of $\text{length}(\pi_1)$ we deduce that the flow f_{π_1} has no positive cycles. We now consider two cases.

Case (i): $i_1 = n$. Then $\pi = \pi_1$, and f_{π_1} is a reachability certificate.

Case (ii): $i_1 < n$. If the terminal segment of π from v_{i_1} to v_n contains a negative cycle, then define $i_2 \geq i_1$ such that v_{i_2} is the last vertex that appears in a negative cycle in π; otherwise let $i_2 = i_1$. Write $\pi_3 = v_{i_2} v_{i_2+1} \dots v_n$. Assume π is chosen, subject to the original choice to minimise $\text{length}(\pi_1)$, such that $\text{length}(\pi_3)$ is minimised. Then π_3 contains only positive cycles; thus from Proposition 7 and the minimality of $\text{length}(\pi_3)$ we deduce that the flow f_{π_3} has no negative cycles. We now consider two sub-cases.

Case (ii)(a): $i_1 = i_2$. Then $\pi = \pi_1 \cdot \pi_3$, and f_{π_1} and f_{π_3} are both reachability certificates.

Case (ii)(b): $i_1 < i_2$. Then write $\pi_2 = v_{i_1} v_{i_1+1} \dots v_{i_2}$. Starting in configuration (v,c), let (v_{i_1}, c_{i_1}) be the configuration of $\mathcal{C}$ after executing π_1, and let (v_{i_2}, c_{i_2}) be the configuration of $\mathcal{C}$ after further executing π_2. By definition of π_2 there is a positive cycle that can be taken from (v_{i_1}, c_{i_1}) and a negative cycle that can be taken to (v_{i_2}, c_{i_2}). Thus f_{π_2} is a reachability certificate and $\pi = \pi_1 \cdot \pi_2 \cdot \pi_3$ is the sequential composition of three paths, each of which yields a reachability certificate.

□

We can now complete the proof of the first main result of the paper.

Theorem 3. *The reachability problem for one-counter automata is in* NP.

Proof. Let $\mathcal{C}$ be a one-counter automaton with configurations (v,c) and (v',c'). If (v',c') is reachable from (v,c) then, by Proposition 8, there is a computation $\pi = \pi_1 \cdot \pi_2 \cdot \pi_3$ from (v,c) to (v',c') such that π_1, π_2 and π_3 each yield reachability

certificates. Moreover we can assume, without loss of generality, that the lengths of π_1, π_2 and π_3 are bounded by 2^P for some polynomial P in $|\mathcal{C}|$ and the bit lengths of c and c'. The bounds on π_1 and π_3 follow from the fact that π_1 has only negative cycles and π_3 has only positive cycles. The bound on π_2 follows from Proposition 2. Thus the reachability certificates corresponding to π_1, π_2 and π_3 all have polynomial size, and, by Corollary 1, can be guessed and verified in polynomial time. □

6 Parametric Counter Automata

In this section we exploit the results developed in Section 5 to show that the reachability problem for parametric one-counter automata can be reduced to the satisfiability problem for a decidable extension of existential Presburger arithmetic.

Let $x_1, \ldots, x_n$ be a set of integer variables. A **linear polynomial** is a polynomial of the form $a_0 + a_1x_1 + \ldots + a_nx_n$, where the a_i are integer coefficients. A **linear constraint** is an inequality of the form $a_0 + a_1x_1 + \ldots + a_nx_n \leq 0$. Define $S \subseteq \mathbb{Z}^n$ to be an ($\mathbb{N}$-)**linear set** if there exist vectors $v_0, v_1, \ldots, v_t \in \mathbb{Z}^n$ such that $S = \{v : v = v_0 + b_1v_1 + \ldots + b_nv_n, b_i \in \mathbb{N}\}$. A **semilinear set** is a finite union of linear sets.

Presburger arithmetic is the first-order theory of the structure $(\mathbb{Z}, <, +, 0, 1)$. It is well-known that the satisfiability problem for Presburger arithmetic is decidable, and that subsets of $\mathbb{Z}^k$ definable by formulas of Presburger arithmetic are effectively semilinear. Adding multiplication to Presburger arithmetic leads to undecidability, as does adding the *divides* predicate $n \mid m$. However Lipshitz [17] gave a decision procedure for the satisfiability problem for the existential fragment of Presburger arithmetic with divisibility. This last result has been used to show the decidability of certain problems concerning systems of quadratic Diophantine equations [10, 11]. We give a simple application of this kind below.

Let $\{y_1, \ldots, y_k\}$ and $\{x_1, \ldots, x_n\}$ be disjoint sets of integer variables. For $1 \leq i \leq k$ let R_i denote the quadratic polynomial $y_iA_i + B_i$, where A_i and B_i are linear polynomials in $x_1, \ldots, x_n$. Furthermore, let P be a subset of $\mathbb{Z}^k$ defined by a formula of Presburger arithmetic. We consider the following problem:
Problem A: Given $R_1, \ldots, R_k$ and P, are there values for $x_1, \ldots, x_n$ and $y_1, \ldots, y_k$ such that $(R_1, \ldots, R_k) \in P$?

Lemma 1. *Problem A is decidable.*

Proof. The proof is by reduction to the satisfiability problem for the existential fragment of Presburger arithmetic with divisibility.

Note that $P \subseteq \mathbb{Z}^k$, being Presburger definable, is effectively semilinear. By case splitting we may assume that P defines a linear set, say $P = \{v : v = v_0 + a_1v_1 + \ldots + a_tv_t, a_i \in \mathbb{N}\}$ where $v_0, \ldots, v_t \in \mathbb{Z}^k$. Thus, introducing new nonnegative integer variables $w_1, \ldots, w_t$, we seek a solution to the following

system of equations

$$\begin{aligned} y_1 A_1 + B_1 &= v_{0,1} + w_1 v_{1,1} + \ldots + w_t v_{t,1} \\ y_2 A_2 + B_2 &= v_{0,2} + w_1 v_{1,2} + \ldots + w_t v_{t,2} \\ &\vdots \\ y_k A_k + B_k &= v_{0,k} + w_1 v_{1,k} + \ldots + w_t v_{t,k} \end{aligned}$$

But this is equivalent to finding a solution to the following formula in Presburger arithmetic with divisibility:

$$\bigwedge_{i=1}^{k} A_i \mid (v_{0,i} + w_1 v_{1,i} + \ldots + w_t v_{t,i} - B_i) \wedge \bigwedge_{i=1}^{t} w_i \geq 0 \,.$$

Remark 1. Note that in Problem A, each variable y_i occurs in a single quadratic polynomial. It immediately follows from a result of Ibarra and Dang [10] that generalising Problem A to allow the same variable y_i to appear in two separate quadratic polynomials leads to an undecidable problem.

6.1 Reachability

Let $\mathcal{C} = (V, E, X, \lambda)$ be a parametric one-counter automaton, and assume for now that $\mathcal{C}$ does not have any zero tests. Recall that the reachability problem asks whether there is a computation between given configurations (v, c) and (v', c') for *some* instantiation of the parameters. By Proposition 8, the existence of such a computation is witnessed by (at most) three reachability certificates. Thus our strategy to show decidability of reachability is to phrase the existence of each of the three types of reachability certificate as an instance of Problem A, with variables representing the parameters. We illustrate the idea for type-1 certificates, the other cases being very similar.

Recall that a type-1 reachability certificate for configurations (v, c) and (v', c') consists of a path flow f from v to v' such that f has no positive cycles, $\mathit{weight}(f) = c' - c$, and there is a decomposition $f = f_0 + \ldots + f_{n-1}$, such that

$$\bigwedge_{j=0}^{n-1} \left(\sum_{i=0}^{j} \mathit{weight}(f_i) \geq -c \right) . \tag{1}$$

In encoding the existence of such an f, let us temporarily assume that the **support** $E_i \stackrel{\text{def}}{=} \{e \in E : f_i(e) > 0\}$ of each flow f_i has been fixed beforehand, subject to the requirement that $f = f_0 + \cdots f_{n-1}$ be a flow decomposition. Thus it only remains to determine the flow along each edge of E_i.

Let the set of edges E have cardinality m. We introduce a set of nonnegative integer variables $Y^{(i)} = \{y_1^{(i)}, y_2^{(i)}, \ldots, y_m^{(i)}\}$ to represent the flow f_i, $0 \leq i < n$. The idea is that each variable represents the flow along a given edge. The flow

conservation conditions on f_i and the requirement that f_i have support E_i can be expressed as a system $S^{(i)}$ of linear constraints on the set of variables $Y^{(i)}$.

We also have a set of integer variables $X = \{x_1, x_2, \ldots, x_k\}$ representing the parameters of $\mathcal{C}$. The requirement that f have no positive cycles can be expressed as a system of linear constraints:

$$A_j \leq 0\,, \qquad j = 0, \ldots, t\,, \tag{2}$$

where A_j is a linear polynomial in the set of variables X, and there is one constraint for each simple cycle of f (exactly which equations need to be written here, which depends on the simple cycles in f, is determined by the supports $E_1, E_2, \ldots, E_{n-1}$.)

The weight of flow f_i can then be expressed as a quadratic expression in the set of variables $X \cup Y^{(i)}$:

$$\mathit{weight}(f_i) = \sum_{j=1}^{m} y_j^{(i)} \alpha_j \qquad [\alpha_j \in \mathbb{Z} \cup X]\,. \tag{3}$$

The next step is to eliminate the system of constraints $S^{(i)}$ by a change of variables. Note that the constraints $S^{(i)}$ on the set of variables $Y^{(i)}$ define a linear set, thus we can introduce a set of nonnegative integer variables $U^{(i)} = \{u_1^{(i)}, u_2^{(i)}, \ldots, u_{l_i}^{(i)}\}$ and linear polynomials $B_j^{(i)}$, $1 \leq j \leq m$, in $U^{(i)}$, such that $(y_1^{(i)}, \ldots, y_m^{(i)})$ satisfies $S^{(i)}$ iff $y_j^{(i)} = B_j^{(i)}$ for some choice of the variables in $U^{(i)}$. Applying this change of variables to Equation (3) and rearranging terms yields

$$\mathit{weight}(f_i) = \sum_{j=1}^{l_i} u_j^{(i)} C_j^{(i)} + D^{(i)}\,, \tag{4}$$

where the $C_j^{(i)}$ and $D^{(i)}$ are linear polynomials in X.

We can now formulate the existence of a type-1 reachability certificate as an instance of Problem A. To this end we introduce a family $R_{i,j}$ of quadratic polynomials over the set of variables $X \cup U^{(i)}$, where $R_{i,j} \stackrel{\text{def}}{=} u_j^{(i)} C_j$ for $0 \leq i < n$ and $1 \leq j \leq l_i$. By (4) the weight of each flow f_i can be written as a linear expression in $D^{(i)}$ and $R_{i,j}$. Thus requirements (1) and (2) can be expressed as a Presburger definable relation P on the A_i, $D^{(i)}$ and $R_{i,j}$, according to the format of Problem A.

Finally, we note that we can drop our assumption of the fixity of the supports $E_1, E_2, \ldots, E_{n-1}$ by case splitting, using the closure of Presburger definable sets under disjunction. Thus we can phrase the existence of a type-1 reachability certificate between two given configurations as an instance of Problem A.

In a similar fashion, the existence of type-2 and type-3 reachability certificates can also be translated into instances of Problem A. Combining with Proposition 8 we derive our second main result:

Theorem 4. *The reachability problem for parametric one-counter automata is decidable.*

References

1. P. A. Abdulla and K. Cerans. Simulation is decidable for one-counter nets (extended abstract). In *CONCUR*, volume 1466 of *LNCS*. Springer, 1998.
2. R. Alur, T.A. Henzinger, and M. Y. Vardi. Parametric real-time reasoning. In *STOC*. ACM, 1993.
3. A. Bouajjani, M. Bozga, P. Habermehl, R. Iosif, P. Moro, and T. Vojnar. Programs with lists are counter automata. In *CAV*, volume 4144 of *LNCS*. Springer, 2006.
4. M. Bozga, R. Iosif, and Y. Lakhnech. Flat parametric counter automata. In *ICALP*, volume 4052 of *LNCS*. Springer, 2006.
5. H. Comon and Y. Jurski. Multiple counters automata, safety analysis and Presburger arithmetic. In *CAV*, volume 1427 of *LNCS*. Springer, 1998.
6. T. Cormen, C. Leiserson, and R. Rivest. *Introduction to Algorithms*. MIT Press and McGraw-Hill, 1990.
7. S. Demri. *Logiques pour la spécification et vérification*. Mémoire d'habilitation, Université Paris 7, 2007.
8. S. Demri and R. Gascon. The effects of bounding syntactic resources on Presburger LTL. In *TIME*. IEEE Computer Society Press, 2007.
9. M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., 1979.
10. O. H. Ibarra and Z. Dang. On two-way finite automata with monotonic counters and quadratic diophantine equations. *Theor. Comput. Sci.*, 312(2-3):359–378, 2004.
11. O. H. Ibarra and Z. Dang. On the solvability of a class of diophantine equations and applications. *Theor. Comput. Sci.*, 352(1):342–346, 2006.
12. O. H. Ibarra, T. Jiang, N. Trân, and H. Wang. New decidability results concerning two-way counter machines and applications. In *ICALP*, volume 700 of *LNCS*. Springer, 1993.
13. P. Jančar, A. Kučera, F. Moller, and Z. Sawa. DP lower bounds for equivalence-checking and model-checking of one-counter automata. *Inf. Comput.*, 188(1):1–19, 2004.
14. A. Kučera. Efficient verification algorithms for one-counter processes. In *ICALP*, volume 1853 of *LNCS*. Springer, 2000.
15. P. Lafourcade, D. Lugiez, and R. Treinen. Intruder deduction for AC-like equational theories with homomorphisms. In *Research Report LSV-04-16*. LSV, ENS de Cachan, 2004.
16. J. Leroux and G. Sutre. Flat counter automata almost everywhere! In *ATVA*, volume 3707 of *LNCS*. Springer, 2005.
17. L. Lipshitz. The diophantine problem for addition and divisibility. *Transactions of the American Mathematical Society*, 235:271–283, 1976.
18. E. W. Mayr. An algorithm for the general petri net reachability problem. In *STOC*, pages 238–246. ACM, 1981.
19. M. Minsky. Recursive unsolvability of Post's problem of "Tag" and other topics in theory of Turing machines. *Annals of Math.*, 74(3), 1961.
20. G. Xie, Z. Dang, and O. H. Ibarra. A solvable class of quadratic diophantine equations with applications to verification of infinite-state systems. In *ICALP*, volume 2719 of *LNCS*. Springer, 2003.