Bisimilarity of one-counter processes is PSPACE-complete

Stanislav Böhm[1⋆], Stefan Göller[2], and Petr Jančar[1]

[1] Techn. Univ. Ostrava (FEI VŠB-TUO), Dept of Computer Science, Czech Republic
[2] Universität Bremen, Institut für Informatik, Germany

Abstract. A one-counter automaton is a pushdown automaton over a singleton stack alphabet. We prove that the bisimilarity of processes generated by nondeterministic one-counter automata (with no ε-steps) is in PSPACE. This improves the previously known decidability result (Jančar 2000), and matches the known PSPACE lower bound (Srba 2009). We add the PTIME-completeness result for deciding regularity (i.e. finiteness up to bisimilarity) of one-counter processes.

1 Introduction

Among the various notions of behavioral equivalences of (reactive) systems, (strong) *bisimilarity* plays an important rôle (cf, e.g., [16]). For instance, various logics can be characterized as the bisimulation-invariant fragment of richer logics. A famous theorem due to van Benthem states that the properties expressible in modal logic coincide with the bisimulation-invariant properties expressible in first-order logic [28]. Similar such characterizations have been obtained for the modal μ-calculus [8] and for CTL* [17]. Another important notion is *weak bisimilarity* that generalizes (strong) bisimilarity by distinguishing ε-moves corresponding to internal behavior. There are numerous further notions of equivalences. For a more detailed treatment of the different behavioral equivalences in the context of concurrency theory, the reader is referred to [4].

The *(weak/strong) bisimilarity problem* consists in deciding if two given states of a given transition system are weakly/strongly bisimilar. On *finite transition systems* both weak and strong bisimilarity is well-known to be complete for deterministic polynomial time [1]. Moreover, on finite transition systems weak bisimilarity can be reduced to strong bisimilarity in polynomial time by computing the transitive closure.

In the last twenty years a lot of research has been devoted to checking behavioral equivalence of infinite-state systems, see [23] for an up-to-date record. In the setting of infinite-state systems, see also [14] for Mayr's classification of infinite-state systems, the situation is less clear. There are numerous classes of infinite-state systems for which *decidability* of bisimilarity is not known. Three such intricate open problems are (i) weak bisimilarity on basic parallel processes (BPP, a subclass of Petri nets), (ii) strong bisimilarity of process algebras (PA), and (iii) weak bisimilarity of basic process algebras (BPA). On the negative side, we mention undecidability of weak bisimilarity of PA by Srba [22]. On the positive side we mention an important result by Sénizergues who

⋆ S. Böhm and P. Jančar are supported by the Czech Ministry of Education, project No. 1M0567.

shows that bisimilarity on equational graphs of finite out degree [19] (a slight generalization of pushdown graphs) is decidable. See also Stirling's unpublished paper [25] for a shorter proof of this, using ideas from concurrency theory. For normed PA processes Hirshfeld and Jerrum prove decidability of strong bisimilarity [7].

When focussing on the *computational complexity* of bisimilarity checking of infinite-state systems for which this problem is decidable, the situation becomes even worse. There are only very few classes of infinite-state systemss for which the precise computational complexity is known. For instance, when coming back to one of the above-mentioned positive results by Sénizergues/Stirling concerning (slight extensions of) pushdown graphs, a primitive recursive upper bound is not yet known. However, EXPTIME hardness of this problem was proven by Kučera and Mayr [13]. As one of the few results on infinite systems where the upper and lower complexity bounds match, we can mention [10] where it is shown that bisimilarity on basic parallel processes is PSPACE-complete.

In this paper we study the computational complexity of deciding strong bisimilarity over processes generated by one-counter automata. One-counter automata are pushdown automata over a singleton stack alphabet. This model has been extensively studied in the verification community; we can name, e.g., [2, 5, 6, 3, 26] as recent works. Weak bisimilarity for one-counter processes is shown to be undecidable in [15], via a reduction from the emptiness problem of Minsky machines.

For strong bisimilarity the third author established decidability in [9], however without providing any precise complexity bounds. In an unpublished article [29] Yen analyses the approach of [9], deriving a triply exponential space upper bound. A PSPACE lower bound for bisimilarity is proven by Srba [24]. This lower bound already holds over one-counter automata that cannot test for zero and whose actions can moreover be restricted to be *visible* (so called *visibly one-counter nets*), i.e. that the label of the action determines if the counter is incremented, decremented, or not modified respectively. For visibly one-counter automata it is proven in [24] that strong bisimilarity is in PSPACE via reduction to the model checking problem of the modal μ-calculus over one-counter processes [20]. For bisimilarity on general one-counter processes, in particular when dropping the visibility restriction, the situation is surely more involved.

Our main result closes the complexity gap for bisimilarity of one-counter processes from above, thus establishing PSPACE-completeness. In a nutshell, we provide a nondeterministic algorithm implementable in polynomial space which generates a bisimulation relation *on-the-fly*. The algorithm uses a polynomial-time procedure which, given a pair $p(m), q(n)$ of processes, either gives a definite answer 'surely bisimilar' or 'surely non-bisimilar', or declares the pair as a *candidate*. For each fixed m there are (only) polynomially many candidates $(p(m), q(n))$, and the algorithm processes each $m = 0, 1, 2, \ldots$ in turn, guessing the bisimilarity status of all respective candidates and verifying the (local) consistency of the guesses. A crucial point is that it is sufficient to stop the processing after exponentially many steps, since then a certain periodicity is guaranteed, which would enable to successfully continue forever.

We also consider the problem of deciding *regularity* (finiteness w.r.t. bisimilarity) which asks if, for a given one-counter process, there is a bisimilar state in some finite system. Decidability of this problem was proven in [9] and according to [24] it follows

from [1] and [21] that the problem is also hard for P. We give a simpler P-hardness proof, but we also show that the regularity problem is in P, thus establishing its P-completeness. It is appropriate to add that Kučera [12] showed a polynomial algorithm deciding bisimilarity between a one-counter process and a (given) finite system state.

The paper is organized as follows. Section 2 contains the basic notions, definitions, and recalls some auxiliary results. Section 3 recalls and enhances some useful notions which were used in [9] and elsewhere. Section 4 contains the crucial technical results, which have enabled to replace the decision algorithm from [9] with a polynomial space algorithm. The algorithm is elaborated in Section 5 and its correctness is shown in Section 6. Section 7 then shows PTIME-completeness of $\sim$-regularity.

2 Preliminaries

$\mathbb{N}$ denotes the set $\{0, 1, 2, \ldots\}$. For a set X, by $|X|$ we denote its cardinality.

Transition systems. A *(labelled) transition system* is a structure $T = (S, \mathsf{A}, \{\stackrel{a}{\longrightarrow} | \ a \in \mathsf{A}\})$, where S is a set of *states*, A a set of *actions*, and $\stackrel{a}{\longrightarrow} \subseteq S \times S$ is a set of a-labeled *transitions*, for each action $a \in \mathsf{A}$. We define $\longrightarrow = \bigcup_{a \in \mathsf{A}} \stackrel{a}{\longrightarrow}$, and prefer to use the infix notation $s_1 \stackrel{a}{\longrightarrow} s_2$ (resp. $s_1 \longrightarrow s_2$) instead of $(s_1, s_2) \in \stackrel{a}{\longrightarrow}$ (resp. $(s_1, s_2) \in \longrightarrow$). T is a *finite transition system* if S and A are finite; we then define the *size* of T as $|T| = |S| + |\mathsf{A}| + \sum_{a \in \mathsf{A}} |\stackrel{a}{\longrightarrow}|$.

Bisimulation equivalence. Let $T = (S, \mathsf{A}, \{\stackrel{a}{\longrightarrow} | \ a \in \mathsf{A}\})$ be a transition system. A binary relation $R \subseteq S \times S$ is a *bisimulation* if for each $(s_1, s_2) \in R$ the following *bisimulation condition* holds:

- for each $s_1' \in S$, $a \in \mathsf{A}$, where $s_1 \stackrel{a}{\longrightarrow} s_1'$, there is some $s_2' \in S$ such that $s_2 \stackrel{a}{\longrightarrow} s_2'$ and $(s_1', s_2') \in R$, and
- for each $s_2' \in S$, $a \in \mathsf{A}$, where $s_2 \stackrel{a}{\longrightarrow} s_2'$, there is some $s_1' \in S$ such that $s_1 \stackrel{a}{\longrightarrow} s_1'$ and $(s_1', s_2') \in R$.

We say that states s_1 and s_2 are *bisimilar*, abbreviated by $s_1 \sim s_2$, if there is a bisimulation R containing (s_1, s_2). Bisimilarity $\sim$ is obviously an equivalence. We also note that the union of bisimulations is a bisimulation, and that $\sim$ is the maximal bisimulation on S. Bisimilarity is naturally defined also between states of different transition systems (by considering their disjoint union).

One-counter automata. A *one-counter automaton* is a tuple $M = (Q, \mathsf{A}, \delta_{=0}, \delta_{>0})$, where Q is a finite nonempty set of *control states*, A is a finite set of *actions*, $\delta_{=0} \subseteq Q \times \{0, 1\} \times \mathsf{A} \times Q$ is a finite set of *zero transitions*, and $\delta_{>0} \subseteq Q \times \{-1, 0, 1\} \times \mathsf{A} \times Q$ is a finite set of *positive transitions*. (There are no ε-steps in M.)
The *size* of M is defined as $|M| = |Q| + |\mathsf{A}| + |\delta_{=0}| + |\delta_{>0}|$. Each one-counter automaton $M = (Q, \mathsf{A}, \delta_{=0}, \delta_{>0})$ defines the transition system $T_M = (Q \times \mathbb{N}, \mathsf{A}, \{\stackrel{a}{\longrightarrow} | \ a \in \mathsf{A}\})$,

where $(q,n) \xrightarrow{a} (q',n+i)$ iff either $n = 0$ and $(q,i,a,q') \in \delta_{=0}$, or $n > 0$ and $(q,i,a,q') \in \delta_{>0}$.
A *one-counter net* is a one-counter automaton, where $\delta_{=0} \subseteq \delta_{>0}$.
A state (q,m) of T_M is also called a *configuration* of M, or a *one-counter process*; we usually write it as $q(m)$. Elements of $\delta_{=0} \cup \delta_{>0}$ are called *transitions*. The notion of a *path* $p(m) \xrightarrow{\sigma} q(n)$, where σ is a *sequence of transitions*, is defined in the natural way. A transition sequence β in $(\delta_{>0})^+$ is called an *elementary cycle* if it induces an elementary cycle in the control state set Q, i.e., if $\beta = (q_1,i_1,a_1,q_2),(q_2,i_2,a_2,q_3),\ldots(q_m,i_m,a_m,q_{m+1})$ where $q_i \neq q_j$ for $1 \leq i < j \leq m$ and $q_{m+1} = q_1$. We note that such a cycle has length at most $|Q|$, and its effect (i.e., the caused change) on the counter value is in the set $\{-|Q|,-|Q|+1,\ldots,|Q|\}$.

Decision problems. We are interested in the following two decision problems.

BISIMILARITY ON OCA

INPUT: A one-counter automaton M with two states $p_0(m_0)$ and $q_0(n_0)$ of T_M, where both m_0 and n_0 are given in binary.
QUESTION: Is $p_0(m_0) \sim q_0(n_0)$?

We say that a *one-counter process* $q(n)$, i.e. a configuration $q(n)$ of a one-counter automaton M, is *$\sim$-regular* (or *finite up to bisimilarity*) if there is a finite transition system with some state s such that $q(n) \sim s$.

$\sim$REGULARITY ON OCA

INPUT: A one-counter automaton M and a state $q(n)$ of T_M (n given in binary).
QUESTION: Is $q(n)$ $\sim$-regular?

Stratified bisimilarity. Given a transition system $T = (S, \mathsf{A}, \{\xrightarrow{a} \mid a \in \mathsf{A}\})$, on S we define the family of i-equivalences, $i \in \mathbb{N}$, $\sim_0 \supseteq \sim_1 \supseteq \sim_2 \supseteq \cdots$ as follows. We put $\sim_0 = S \times S$, and we have $s_1 \sim_{i+1} s_2$ if the following two conditions hold:

- for each $s_1' \in S$, $a \in \mathsf{A}$, where $s_1 \xrightarrow{a} s_1'$, there is some $s_2' \in S$ such that $s_2 \xrightarrow{a} s_2'$ and $s_1' \sim_i s_2'$;
- for each $s_2' \in S$, $a \in \mathsf{A}$, where $s_2 \xrightarrow{a} s_2'$, there is some $s_1' \in S$ such that $s_1 \xrightarrow{a} s_1'$ and $s_1' \sim_i s_2'$.

The following proposition is an instance of the result for image finite systems [16].

Proposition 1. *On states of T_M we have $\sim = \bigcap_{i \geq 0} \sim_i$.*

2.1 Some useful observations

The next proposition captures the locality of the bisimulation condition for one-counter automata, implied by the fact that the counter value can change by at most 1 in a move.

Proposition 2. *Given a one-counter automaton $M = (Q, \mathsf{A}, \delta_{=0}, \delta_{>0})$ and a relation $\mathcal{R} \subseteq (Q \times \mathbb{N}) \times (Q \times \mathbb{N})$, for checking if a pair $(p(m), q(n)) \in \mathcal{R}$ satisfies the bisimulation condition it suffices to know the restriction of $\mathcal{R}$ to the set* $\text{NEIGHBOURS}(m,n) = \{\, (p'(m'), q'(n')) \mid |m' - m| \leq 1, |n' - n| \leq 1 \,\}$.

Standard partition arguments [11, 18] imply the following proposition for *finite* systems.

Proposition 3. *Given a finite transition system $F = (Q, \mathsf{A}, \{\stackrel{a}{\longrightarrow} \mid a \in \mathsf{A}\})$, where $k = |Q|$, we have $\sim_{k-1} = \sim_k = \sim$ on Q. Moreover, (the partition of Q corresponding to) $\sim$ can be computed in polynomial time.*

3 Underlying finite automaton and the set INC

Most of the notions, claims and ideas in this section appeared in [9] and elsewhere; nevertheless, we present (and extend) them in a concise self-contained way.

If the context does not indicate otherwise, in what follows we (often implicitly) assume a fixed one-counter automaton $M = (Q, \mathsf{A}, \delta_{=0}, \delta_{>0})$, using k for $|Q|$. We start by observing that if the counter value is large, then M behaves, for a long time, like a (nondeterministic) finite automaton. By F_M we denote the *finite transition system underlying* M; we put $F_M = (Q, \mathsf{A}, \{\stackrel{a}{\longrightarrow} \mid a \in \mathsf{A}\})$, where $\stackrel{a}{\longrightarrow} = \{(q_1, q_2) \in Q \times Q \mid \exists i : (q, i, a, q') \in \delta_{>0}\}$. ($F_M$ thus behaves as if the counter is positive, ignoring the counter changes.) In what follows, $p, q, r \in Q$ are viewed as control states of M or as states of F_M, depending on context. Our observation is formalized by the next proposition (which is obvious, by induction on m).

Proposition 4. *If $m' \geq m$ then $p(m') \sim_m p$.*
(Here $p(m')$ is a state of T_M, whereas p is a state of F_M.)

This implies, e.g., that if $p \not\sim q$ (i.e., $p \not\sim_k q$ by Proposition 3) and $m, n \geq k$, then $p(m) \not\sim_k q(n)$ (and thus $p(m) \not\sim q(n)$), since $p(m) \sim_k p$, $q(n) \sim_k q$ and $\sim_k$ is an equivalence. If $p \sim q$ then we can have $p(m) \not\sim q(n)$, due to the possibility of reaching zero. For making this more precise, we define the following set

$$\text{INC} = \{\, r(\ell) \mid \forall q \in Q : r(\ell) \not\sim_k q \,\}\,.$$

The configurations in INC are *incompatible with* F_M in the sense that they are not bisimilar up to k moves with any state of F_M.

Proposition 5. *If $r(\ell) \in$ INC then $\ell < k$. Moreover,* INC *can be constructed in polynomial time.*

Proof. If $\ell \geq k$ then $r(\ell) \sim_k r$, and thus $r(\ell) \notin$ INC. To construct INC, we can start with the set containing all k^3 pairs $(r(\ell), q)$, where $\ell < k$; all such pairs belong to $\sim_0$. We then delete the pairs not belonging to $\sim_1$, then those not belonging to $\sim_2$, etc., until $\sim_k$. The configurations $r(\ell)$ for which no pair $(r(\ell), q)$ survived, are in INC. (This process can be done simultaneously for the pairs (p, q) in F_M; we also use the fact $r(k) \sim_k r$.) □

The arguments of the previous proof also induce the following useful proposition.

Proposition 6. *The question if $p(m) \sim_k q(n)$ can be decided in polynomial time.*

We note that if $p(m) \in \mathrm{INC}$ and $q(n) \notin \mathrm{INC}$ then $p(m) \not\sim q(n)$ (in fact, $p(m) \not\sim_k q(n)$). More generally, if two one-counter processes are bisimilar then they must agree on the distance to INC; this is formalized by the next lemma. We define

$$\mathrm{dist}(p(m))$$

as the length of the shortest transition sequence σ such that $p(m) \xrightarrow{\sigma} \mathrm{INC}$ (i.e., $p(m) \xrightarrow{\sigma} r(\ell)$ for some $r(\ell) \in \mathrm{INC}$); we put $\mathrm{dist}(p(m)) = \omega$ if there is no such sequence, i.e., when INC is unreachable, denoted $p(m) \not\rightarrow^* \mathrm{INC}$.

Lemma 7. *If $p(m) \sim q(n)$ then dist$(p(m)) =$ dist$(q(n))$.*

Proof. For the sake of contradiction, suppose that $p(m) \sim q(n)$ and $d = \mathrm{dist}(p(m)) < \mathrm{dist}(q(n))$, for the least d; necessarily $d > 0$, since we cannot have $p(m) \in \mathrm{INC}$, $q(n) \notin \mathrm{INC}$. Thus there is a move $p(m) \xrightarrow{a} p'(m')$ with $\mathrm{dist}(p'(m')) = d-1$, which must be matched by some $q(n) \xrightarrow{a} q'(n')$ where $p'(m') \sim q'(n')$. Necessarily $d-1 = \mathrm{dist}(p'(m')) < \mathrm{dist}(q'(n'))$, which contradicts the minimality of d. □

The next lemma clarifies the opposite direction in the case of infinite distances.

Lemma 8. *If dist$(p(m)) = \omega$ then $p(m) \sim r$ for some $r \in Q$. Thus if dist$(p(m)) =$ dist$(q(n)) = \omega$ then $p(m) \sim q(n)$ iff there is some $r \in Q$ such that $p(m) \sim_k r \sim_k q(n)$.*

Proof. If $\mathrm{dist}(p(m)) = \omega$, i.e. $p(m) \not\rightarrow^* \mathrm{INC}$, then in particular $p(m) \notin \mathrm{INC}$, and there is thus $r \in Q$ such that $p(m) \sim_k r$. We can easily check that

$$R = \{\, (p(m), r) \mid p(m) \sim_k r,\ p(m) \not\rightarrow^* \mathrm{INC} \,\}$$

is a bisimulation: if $p(m) \xrightarrow{a} p'(m')$ and $r \xrightarrow{a} r'$ where $p'(m') \sim_{k-1} r'$, then $p'(m') \not\rightarrow^* \mathrm{INC}$ and the fact $p'(m') \notin \mathrm{INC}$ implies that $r'' \sim_k p'(m') \sim_{k-1} r'$ for some $r'' \in Q$; hence $r' \sim_{k-1} r''$ and thus $r' \sim_k r''$ (by Proposition 3), which means $p'(m') \sim_k r'$. □

In the next section we look in more detail at the function $dist(p(m))$, which provides a useful constraint on bisimilar pairs. But before that, we partition the set $(Q \times \mathbb{N}) \times (Q \times \mathbb{N})$ into three categories. We say that a pair $(p(m), q(n))$ is

- *surely-positive* if $dist(p(m)) = dist(q(n)) = \omega$ and $p(m) \sim_k q(n)$ (and thus surely $p(m) \sim q(n)$, by Lemma 8),
- *surely-negative* if $p(m) \not\sim_k q(n)$ or $dist(p(m)) \neq dist(q(n))$ (and thus surely $p(m) \not\sim q(n)$),
- *candidate* otherwise, i.e., if $p(m) \sim_k q(n)$ and $dist(p(m)) = dist(q(n)) < \omega$.

By SUREPOS we denote the set of all surely-positive pairs, and we note the following obvious proposition.

Proposition 9. SUREPOS *is a bisimulation.*

It will be also useful to view the set CAND of all candidate pairs as the union

$$\mathrm{CAND} = \mathrm{CAND}_0 \cup \mathrm{CAND}_1 \cup \mathrm{CAND}_2 \cup \cdots$$

where CAND_i contains the *candidate pairs at level* i, i.e. the pairs $(p(m), q(n)) \in$ CAND with $m = i$.

4 Distance to INC

In this section we look at the distance function $dist(p(m))$ in more detail (Lemma 10) and derive some consequences (Lemma 11) which will be useful for the design and analysis of our later algorithm.

We start with sketching some intuition which is then formalized in Lemma 10. To reach INC from $p(m)$ most quickly, for a large m, one uses a suitable prefix arriving at a 'most effective' elementary cycle $q(-) \xrightarrow{\beta} q(-)$ (which decreases the counter by $k = |Q|$ at most), let us call it a *d-cycle*, then repeats the d-cycle sufficiently many times, and finishes with a suffix arriving at INC. It is not difficult to anticipate that one can bound the length (and thus also the counter change) of the prefix and the suffix by a (small degree) polynomial pol(k). We now state the lemma. For technical reasons, we do not require explicitly that the d-cycle is elementary; it is sufficient to bound its length by k.

Lemma 10. *There is a polynomial pol* : $\mathbb{N} \to \mathbb{N}$ *(independent of* M*) such that for any* $p(m)$ *with* $dist(p(m)) < \omega$ *there is a shortest path* $p(m) \xrightarrow{\sigma}$ INC *with the transition sequence* σ *of the form* $\sigma = \alpha\beta^i\gamma$ *where* $length(\alpha\gamma) \leq$ *pol*(k) *and* β *is a decreasing cycle of length* $\leq k$.

Proof. To give a complete formal proof requires some technical work. Since the essence of the claim is not original and similar technical results appear in the previous works on one-counter automata, we do not provide a self-contained proof, but we use Lemma 2 from an older paper [27]; in our notation, this lemma is formulated as follows:

> **Claim.** If there is a positive path (using positive transitions) from $p(m)$ to $q(n)$ and $m-n \geq k^2$ and $n \geq k^2$ then there is a shortest path $p(m) \xrightarrow{\sigma} q(n)$ such that $\sigma = \alpha\beta^i\gamma$ where $length(\alpha\gamma) < k^2$ and $length(\beta) \leq k$.

(Although [27] studies *deterministic* one-counter automata, the lemma obviously applies to our nondeterministic case as well, since we can view the transitions themselves as the actions.) We note that if $m-n \geq k^2 + k$ then β is necessarily a decreasing cycle ($i \geq 2$ in this case). It is also clear that the (shortest) path $p(m) \xrightarrow{\sigma} q(n)$ in the Claim does not visit any $q'(n')$ with $n' \geq m + k^2 + k$; we say that the path moves in the "$< (m+k^2+k)$ area" (note that the prefix α moves in the "$< (m+k^2)$ area" and the suffix γ moves in the "$< (n+k^2)$ area").

Recalling that $\ell < k$ for each $r(\ell) \in$ INC, we note that any shortest path $p(m) \xrightarrow{\sigma}$ INC either moves in the "$< k^2$-area", in which case its length is bounded

by k^3 (since no configuration is visited twice), or can be presented in the form $p(m) \xrightarrow{\sigma_1} q_1(k^2) \xrightarrow{\sigma_2} q_2(k^2) \xrightarrow{\sigma_3} \cdots \xrightarrow{\sigma_m} q_m(k^2) \xrightarrow{\sigma_{m+1}}$ INC where $1 \leq m \leq k$ and $q_1(k^2), q_2(k^2), \ldots, q_m(k^2)$ are all configurations on the path which have the counter value k^2. By the above considerations, the segment $q_i(k^2) \xrightarrow{\sigma_{i+1}}$ moves in the "$< (3k^2 + k)$ area", and its length is thus bounded by $k \cdot (3k^2{+}k)$. The segment $p(m) \xrightarrow{\sigma_1} q_1(k^2)$ either moves in the "$< 2k^2$ area", in which case its length is bounded by $2k^3$, or it can be written $p(m) \xrightarrow{\sigma_1'} p'(m') \xrightarrow{\sigma_2'} q_1(k^2)$ where $m' \geq 2k^2$ and σ_1' (which might be empty) is bounded by $2k^3$. The statement of our Lemma thus follows from the above Claim applied to the segment $p'(m') \xrightarrow{\sigma_2'} q_1(k^2)$. □

The next lemma lists some important consequences. A main point is to clarify the distribution of the set CAND. Informally speaking, the candidate pairs are contained inside polynomially many linear belts, each belt having a rational slope, being a fraction of polynomially bounded integers, as well as a polynomially bounded (vertical) thickness.

Remark. It is helpful to think in geometrical notions. Every relation $\mathcal{R} \subseteq (Q \times \mathbb{N}) \times (Q \times \mathbb{N})$ can be viewed as a 'coloring' $\chi_{\mathcal{R}} : Q \times Q \times \mathbb{N} \times \mathbb{N} \to \{\bullet, \circ\}$; for each $p, q \in Q$ it prescribes a black-white coloring of the plane (grid) $\mathbb{N} \times \mathbb{N}$. This was more formalized in [9]; here we just informally use Figure 1.

Lemma 11.

1. *There is a polynomial-time algorithm computing $dist(p(m))$ for any p, m; here the size of the input is $|M| + \log m$ (m is written in binary).*
2. *If $dist(p(m)) < \omega$ then*

$$dist(p(m)) = \frac{c_1}{c_2}(m + d_1) + d_2 = \frac{c_1}{c_2}m + \psi$$

   for some integers $0 \leq c_1 \leq k$, $1 \leq c_2 \leq k$, $|d_1| \leq pol_1(k)$, $0 \leq d_2 \leq pol_1(k)$ where pol_1 is a polynomial (independent of M); the values c_1, c_2, d_1, d_2 generally depend on p, m.
   Moreover, for the rational number $\psi = \frac{c_1}{c_2}d_1 + d_2$ we have $|\psi| \leq (k{+}1) \cdot pol_1(k)$.
3. *If $dist(p(m)) = dist(q(n)) < \omega$ then*

$$n = \rho \cdot m + \xi$$

   where (the slope) ρ is either 0 or of the form $\frac{c_1 c_2'}{c_2 c_1'}$, for $c_1, c_2, c_1', c_2' \in \{1, 2, \ldots, k\}$, and $|\xi|$ is bounded by a polynomial $pol_2(k)$.
   (This formalizes the above announced polynomially many belts, with the vertical thickness $1 + 2 \cdot pol_2(k)$.)
4. *There is a polynomial pol_4 such that for each $m \geq pol_4(k)$ we have $\rho_1 \cdot m + pol_2(k) + 1 < \rho_2 \cdot (m{-}1) - pol_2(k)$, where $\rho_1 < \rho_2$ are (different) slopes from Point 3, pol_2 also being taken from there.*
   (I.e., for levels $m \geq pol_4(k)$ the belts are separated, in the sense that no two pairs from different belts are neighbours.)

5. *There is a polynomial-time algorithm which, given i (in binary), computes the set* CAND_i *of all* candidate pairs at level i *(all pairs $(p(i), q(n))$ such that $p(i) \sim_k q(n)$ and $dist(p(i)) = dist(q(n)) < \omega$). We have $|\mathrm{CAND}_i| \leq pol_3(k)$ for a polynomial pol_3.*

6. *If Δ is a multiple of the effects of all decreasing cycles of length $\leq k$ (the absolute values of the effects are in the set $\{1, 2, \ldots, k\}$) then for each $m \geq k + pol(k)$, where pol is taken from Lemma 10, we have:*

$$p(m) \to^* \mathrm{INC} \quad \textit{iff} \quad p(m + \Delta) \to^* \mathrm{INC}.$$

7. *If $m, n \geq k + pol(k)$ then*

$$(p(m), q(n)) \in \textsc{SurePos} \Leftrightarrow \forall i, j \in \mathbb{N} : (p(m + i\Delta), q(n + j\Delta)) \in \textsc{SurePos}$$

(where pol and Δ are as in Point 6).

Proof. Point 1. By Lemma 10 we know that a shortest path $p(m) \xrightarrow{\sigma} \mathrm{INC}$ (if there is any) is of the form

$$p(m) \xrightarrow{\alpha} q(m{+}e_1) \xrightarrow{\beta} q(m{+}e_1{-}c_2) \xrightarrow{\beta} q(m{+}e_1{-}2c_2) \xrightarrow{\beta} \cdots$$
$$\cdots \xrightarrow{\beta} q(\ell{-}e_2{+}c_2) \xrightarrow{\beta} q(\ell{-}e_2) \xrightarrow{\gamma} r(\ell) \in \mathrm{INC}$$

where e_1 is the effect (the counter change) of the prefix α, c_2 is the absolute value of the effect of the d-cycle β, and e_2 is the effect of the suffix γ; we put $c_1 = length(\beta)$, $c_3 = length(\alpha)$, $c_4 = length(\gamma)$. Let us recall that $0 \leq c_2 \leq c_1 \leq k$ and that the absolute values of other integers are bounded by $\mathrm{pol}(k)$ from Lemma 10.
(Independently of p, m,) we thus have polynomially many possibilities (in k) for the tuple $q, e_1, c_1, c_2, c_3, c_4, e_2, r, \ell$; these possible tuples can be processed in turn. For each tuple we can check if $(m{+}e_1){-}(\ell{-}e_2)$ is divisible by c_2 and then verify if the tuple is realizable by some appropriate α, β, γ; this verification is done by using straightforward graph reachability algorithms. (Regarding the d-cycle, it is sufficient to verify the realizability of the first segment $q(m{+}e_1) \longrightarrow q(m{+}e_1{-}c_2)$ and of the final segment $q(\ell{-}e_2{+}c_2) \longrightarrow q(\ell{-}e_2)$.) With each realizable tuple we associate the value $c_3 + c_4$ when $c_2 = 0$ and $c_3 + c_4 + \frac{c_1}{c_2}\big((m{+}e_1){-}(\ell{-}e_2)\big)$ when $c_2 > 0$. We associate ω with each non-realizable tuple. The value $dist(p(m))$ is obviously the minimal value associated with the above tuples.

Point 2. This follows immediately from the analysis in the proof of Point 1. (Since $d_2 = c_3 + c_4$, $d_1 = e_1{-}\ell{+}e_2$, it suffices to take $\mathrm{pol}_1(k) = k + \mathrm{pol}(k)$, for pol from Lemma 10. The consequence for ψ is obvious.)

Point 3. From $dist(p(m)) = \frac{c_1}{c_2}m + \psi = \frac{c'_1}{c'_2}n + \psi' = dist(q(n))$, we derive $n = \frac{c_1/c_2}{c'_1/c'_2}m + \frac{\psi - \psi'}{c'_1/c'_2}$. If $c_1 = 0$ or $c'_1 = 0$ then $dist(p(m)) = dist(q(n)) \leq (k{+}1) \cdot \mathrm{pol}_1(k)$, and thus $n < k + (k{+}1) \cdot \mathrm{pol}_1(k)$ (and we can put $\rho = 0$). We can thus take $\mathrm{pol}_2(k) = 2 \cdot (k{+}1) \cdot \mathrm{pol}_1(k) \cdot k$.

Point 4. Recalling the slopes from Point 3, we note that $\rho_1 {<} \rho_2$ implies $\rho_2 \geq \rho_1 {+} \frac{1}{k^4}$. Since $\rho_1 \leq k^2$, it is sufficient to have $\mathrm{pol}_2(k) + 1 < \frac{1}{k^4}m - k^2 - \frac{1}{k^4} - \mathrm{pol}_2(k)$.

Point 5. Given i, for each $p \in Q$ in turn we compute $z = dist(p(i))$ and all polynomially many n such that $\frac{c_1}{c_2}(n{+}d_1){+}d_2 = z$, where c_1, c_2, d_1, d_2 satisfy the constraints

from Point 2. For each such n and each $q \in Q$ we check if $(p(i), q(n)) \in \mathrm{CAND}_i$, i.e., if $dist(q(n)) = z$ and $p(i) \sim_k q(n)$; Point 1 and Proposition 6 show that this can be done in polynomial time.

Point 6. Since $m \geq k + \mathrm{pol}(k)$, the length of (each) σ such that $p(m) \xrightarrow{\sigma}$ INC is greater than $\mathrm{pol}(k)$. Increasing or decreasing the number of repeating the d-cycle does the job.

Point 7. From Point 6 we know that for $m \geq k+\mathrm{pol}(k)$ we have $dist(p(m)) = \omega$ iff $dist(p(m+i\Delta)) = \omega$ for all $i \in \mathbb{N}$. Thus for $m, n \geq k+\mathrm{pol}(k)$ we have $p(m+i\Delta) \sim_k q(n + j\Delta)$ and $dist(p(m + i\Delta)) = dist(q(n + j\Delta)) = \omega$ if and only if $p \sim_k q$ and $dist(p(m)) = dist(q(n)) = \omega$.

□

5 A polynomial space algorithm

The next lemma follows from Lemma 11, Point 1, and Proposition 6.

Lemma 12. *There is a polynomial-time algorithm which, given (M and) a pair $(p(m), q(n))$, decides if the pair is in* SurePos*, or in* CAND*, or is surely-negative.*

We might be tempted to try to resolve the question of bisimilarity of the candidate pairs by looking for additional polynomially checkable conditions. But the PSPACE-hardness result for (visibly) one-counter processes [24] discourages us from doing so; we should be satisfied with solving our problem in polynomial space. Thus a *nondeterministic* algorithm working in polynomial space is sufficient (since PSPACE=NPSPACE by Savitch's Theorem). We start with noting the following two obvious propositions; this will give rise to a main algorithmic idea.

Proposition 13. *For a candidate pair $(p_0(m_0), q_0(n_0)) \in$ CAND we have:*
$p_0(m_0) \sim q_0(n_0)$ iff there is a subset $B \subseteq$ CAND such that
$(p_0(m_0), q_0(n_0)) \in B$ and $B \cup$ SurePos is a bisimulation.

The following (infinite) algorithm builds a certain $B \subseteq$ CAND as the union of (nondeterministically chosen) sets $B_0 \subseteq \mathrm{CAND}_0$, $B_1 \subseteq \mathrm{CAND}_1$, $B_2 \subseteq \mathrm{CAND}_2$, ..., while checking the bisimulation condition for their elements on the fly (recall the locality captured by Proposition 2). If its computation does not fail, then it is infinite and the respective set $B \subseteq$ CAND (which would result as the limit) satisfies that $B \cup$ SurePos is a bisimulation.

- We start with putting $m = 0$, compute the set CAND_0 and (nondeterministically) choose a set $B_0 \subseteq \mathrm{CAND}_0$.
- Then we successively process $m = 0, 1, 2 \ldots$, where processing m means the following:
  - Compute CAND_{m+1} (recall Point 5 of Lemma 11) and (nondeterministically) choose $B_{m+1} \subseteq \mathrm{CAND}_{m+1}$.
  - Verify that (each pair in) B_m is (locally) correct, using B_{m-1} (when $m > 0$) and B_{m+1}, and the polynomial procedure deciding membership in SurePos (cf. Lemma 12).

- (If B_m is not correct, the computation fails.)

If we force the algorithm to include the input pair $(p_0(m_0), q_0(n_0))$ into B_{m_0} then an infinite run is possible if and only if $p_0(m_0) \sim q_0(n)$. We also note that it is sufficient for the algorithm to keep only the current number m, and the sets B_{m-1} (if $m > 0$), B_m, B_{m+1} in memory. (By Point 5 of Lemma 11 this consists of at most $3 \cdot pol_3(k)$ pairs, while the bit-size of the numbers is polynomial in k and in the bit-size of m, i.e. in $\log m$.)

A final crucial point is that the algorithm, getting $p_0(m_0), q_0(n_0)$ in the input, will halt (answering $p_0(m_0) \sim q_0(n_0)$) after it has successfully processed the following levels.

$$m = 0, 1, 2, \ldots, z \text{ where } z = m_0 + \text{pol}_4(k) + 2^{\text{pol}_5(k)} \cdot 2^{3k \log k} \quad (1)$$

Here pol_4 is from Point 4 of Lemma 11, and we put $\text{pol}_5(k) = 2 \cdot k^2 \cdot (1 + 2 \cdot \text{pol}_2(k))$, where pol_2 is from Point 3 of Lemma 11. With this halting condition, the algorithm obviously runs in polynomial space (when given M and a pair $(p_0(m_0), q_0(n_0))$). What remains to show is the correctness of the halting condition.

6 Correctness of (the halting condition of) the algorithm

Recall Points 3 and 4 of Lemma 11; the candidate pairs are contained inside polynomially many linear belts with vertical thickness $(1 + 2 \cdot \text{pol}_2(k))$, which are separated for $m \geq pol_4(k)$.

Informally speaking, if the algorithm (successfully) processes sufficiently many (exponentially many) numbers m after processing m_0, then the pigeonhole principle guarantees that a certain 'pumpable' segment appears inside each belt (this is visualized in Figure 1). At that time we are guaranteed that the relation

$$\mathcal{R} = \{(p(m), q(n)) \in B_m \cup \textsc{SurePos} \mid m \leq m_0\}$$

can be extended with certain pairs $(p'(m'), q'(n'))$, with $m' > m_0$, so that the resulting relation is a bisimulation. (These pairs $(p'(m'), q'(n'))$, $m' > m_0$, may differ from those which were actually included in $B_{m'}$ by the algorithm.) We now make this informal argument more precise.

Suppose that our algorithm successfully halts for the input pair $(p_0(m_0), q_0(n_0))$, and consider the following subsequence of the sequence (1).

$$m'_0, m'_0 + \Delta^3, m'_0 + 2\Delta^3, m'_0 + 3\Delta^3, \ldots, m'_0 + 2^{\text{pol}_5(k)}\Delta^3 \quad (2)$$

where $m'_0 = \max\{m_0, \text{pol}_4(k)\}$ and $\Delta = k!$; hence $\Delta \leq k^k$, and so $\Delta^3 \leq 2^{3k \log k}$.

Remark. We have chosen Δ so that Points 6 and 7 of Lemma 11 can be applied.

The chosen period Δ^3 has the following useful property. We are guaranteed that $\rho\Delta^3$ is a multiple of Δ for each slope $\rho = \frac{c_1 c'_2}{c_2 c'_1}$ ($c_1, c_2, c'_1, c'_2 \in \{1, 2, \ldots, k\}$) from Point 3 of Lemma 11; by Point 7 of Lemma 11 we thus also get for each $m \geq \text{pol}_4(k)$:

$$(p(m), q(n)) \in \textsc{SurePos} \Leftrightarrow \forall i \in \mathbb{N} : (p(m + i\Delta^3), q(n + i\rho\Delta^3)) \in \textsc{SurePos}\,. \quad (3)$$

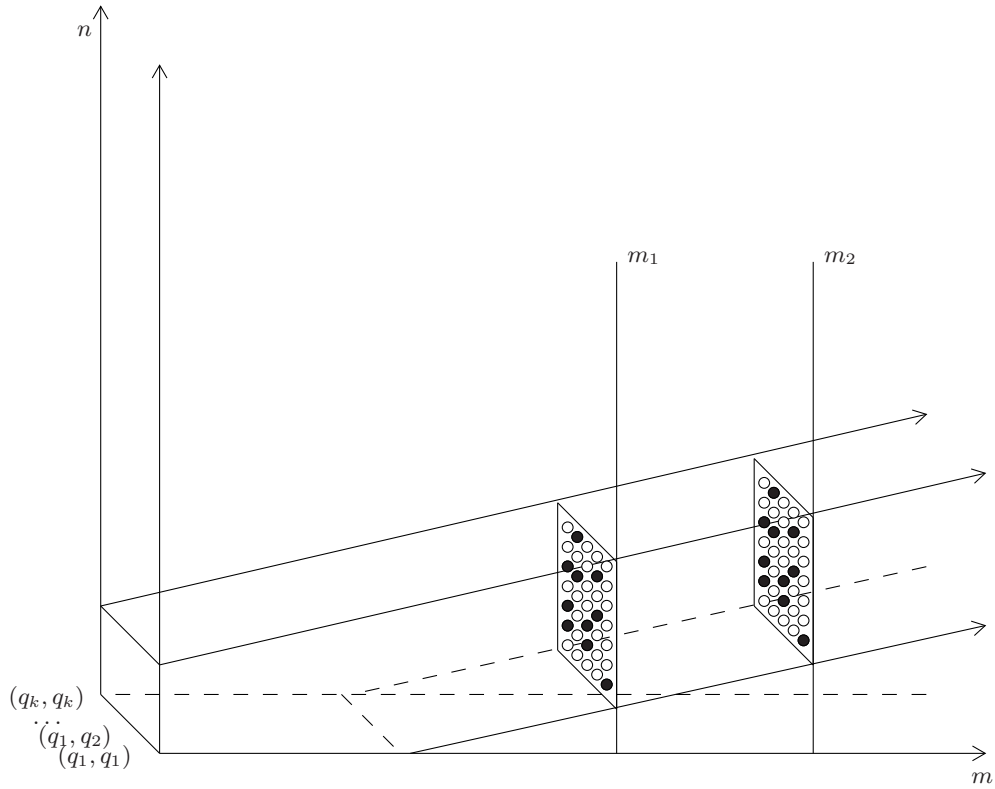

Fig. 1. Two isomorphic belt cuts in a coloring

(In the proof of Lemma 11, we have actually derived pol_4 satisfying $\text{pol}_4(k) \geq k+\text{pol}(k)$. But any polynomial pol_4 satisfying Point 4 could be replaced with a bigger one to satisfy also $\text{pol}_4(k) \geq k+\text{pol}(k)$ anyway.)

For a relation $\mathcal{R} \subseteq (Q \times \mathbb{N}) \times (Q \times \mathbb{N})$ and a belt, identified with its slope ρ from Point 3 of Lemma 11, we define the $\mathcal{R}$-cut of the belt ρ at level m as

$$\text{CUT}^{\rho}_{m}(\mathcal{R}) = \{\, (p(m), q(n)) \in \mathcal{R} \mid \rho m - \text{pol}_2(k) \leq n \leq \rho m + \text{pol}_2(k) \,\}.$$

Figure 1 illustrates two cuts $\text{CUT}^{\rho}_{m_1}(\mathcal{R})$, $\text{CUT}^{\rho}_{m_2}(\mathcal{R})$ (the black points representing elements of $\mathcal{R}$, the white points being non-elements); the depicted cuts are 'the same' in the sense that one arises by shifting the other.

Our choice of the subsequence (2) guarantees a repeat of a '2-thick cut':

Proposition 14. *For every $\mathcal{R}$ and 'belt' ρ there are m_1, m_2 in (2), where $m_1 < m_2$, $m_2 = m_1 + c\Delta^3$, such that*

- $(p(m_1), q(n)) \in \text{CUT}^{\rho}_{m_1}(\mathcal{R}) \Leftrightarrow (p(m_2), q(n + \rho c\Delta^3)) \in \text{CUT}^{\rho}_{m_2}(\mathcal{R})$,
- $(p(m_1+1), q(n)) \in \text{CUT}^{\rho}_{m_1+1}(\mathcal{R}) \Leftrightarrow (p(m_2+1), q(n+\rho c\Delta^3)) \in \text{CUT}^{\rho}_{m_2+1}(\mathcal{R})$.

Proof. We first note that our choice of Δ also guarantees that $\rho c\Delta^3$ is integer. Describing $\text{CUT}^{\rho}_{m}(\mathcal{R})$ and $\text{CUT}^{\rho}_{m+1}(\mathcal{R})$ (for any m) obviously amounts to determine a

(black-point) subset of a set with (at most) $2 \cdot k^2 \cdot (1 + 2 \cdot \mathrm{pol}_2(k))$ elements; this is how we defined $\mathrm{pol}_5(k)$ in the halting condition of our algorithm (cf. (1)). There are $2^{\mathrm{pol}_5(k)}$ such subsets; thus the claim follows by the pigeonhole principle. □

Our aim is to define some relation $\mathcal{R}'$ so that $\mathcal{R}' \cup \textsc{SurePos}$ is a bisimulation and it coincides with $B \cup \textsc{SurePos}$ for the pairs $(p(m), q(n))$ with $m \leq m_0$; the set B consists of the candidate pairs included by (the successfully halting computation of) our algorithm into B_m, for $m = 0, 1, 2, \ldots, z$ as in (1).

Let us now consider a particular belt ρ. Let m_1, m_2, where $m_1 < m_2 = m_1 + c\Delta^3$, be the levels guaranteed by Proposition 14 for the relation $\mathcal{R} = B \cup \textsc{SurePos}$. Inside the belt ρ, the suggested $\mathcal{R}'$ will coincide with $\mathcal{R}$ for all levels $m \leq m_2{+}1$. For all levels $m = m_2{+}2, m_2{+}3, m_2{+}4, \ldots$, we define $\mathcal{R}'$ inside the belt ρ by the following inductive definition: for each m, n, where $m > m_2{+}1$ and $|n - \rho m| \leq \mathrm{pol}_2(k)$:

$$(p(m), q(n)) \in \mathcal{R}' \text{ iff } (p(m{-}c\Delta^3), q(n{-}\rho c\Delta^3)) \in \mathcal{R}'.$$

We note that this condition is, in fact, satisfied also for $m \in \{m_2, m_2{+}1\}$, due to our choice of m_1, m_2. We get the whole $\mathcal{R}'$ after having defined it inside all belts.

Proposition 15. *$\mathcal{R}' \cup \textsc{SurePos}$ is a bisimulation.*

Proof. Suppose there is a pair $(p(m), q(n)) \in \mathcal{R}' \cup \textsc{SurePos}$ which does not satisfy the bisimulation condition (which is determined by the restriction to $\textsc{neighbours}(m, n)$; recall Proposition 2). We take such a pair with the least m. It is clear that $(p(m), q(n)) \notin \textsc{SurePos}$ (recall Proposition 9); moreover, the restriction of $\mathcal{R}' \cup \textsc{SurePos}$ to $\textsc{neighbours}(m, n)$ cannot be the same as for $B \cup \textsc{SurePos}$ (where the algorithm verified the bisimulation condition). Hence (m, n) lies in a belt ρ, and $m \geq m_2{+}1$ for the respective $m_2 = m_1{+}c\Delta^3$. Then the pair $(p(m{-}c\Delta^3), q(n{-}\rho c\Delta^3))$ belongs to $\mathcal{R}'$ and satisfies the bisimulation condition; moreover, this pair enables the same transitions as the pair $(p(m), q(n))$. So there must be some $(p'(m'), q'(n')) \in \textsc{neighbours}(m, n)$ such that $(p'(m'), q'(n')) \notin \mathcal{R}' \cup \textsc{SurePos}$ and $(p'(m'{-}c\Delta^3), q'(n'{-}\rho c\Delta^3)) \in \mathcal{R}' \cup \textsc{SurePos}$. But this contradicts the definition of $\mathcal{R}'$ or the equivalence (3). □

Our halting condition is thus correct, and we have proved:

Theorem 16. *There is a polynomial space algorithm which, given a one-counter automaton M and a pair $p_0(m_0), q_0(n_0)$, decides if $p_0(m_0) \sim q_0(n_0)$.*

Remark. As in [9], we could derive that the bisimilarity $\sim$ (i.e., the maximal bisimulation) is 'belt-regular'. Our results here show that a natural (finite) description of this (semilinear) relation can be written in exponential space.

7 $\sim$-Regularity

We can easily derive the next lemma, which tells us that $p(m)$ is not $\sim$-regular iff it allows to reach states with arbitrarily large finite distances to INC.

Lemma 17. *Given $p(m)$ for a one counter automaton M, $p(m)$ is not $\sim$-regular iff for any $d \in \mathbb{N}$ there is $q(n)$ such that $p(m) \to^* q(n)$ and $d \leq dist(q(n)) < \omega$.*

The next proposition gives a more convenient characterization.

Proposition 18. *$p(m)$ is not $\sim$-regular iff $p(m) \to^* q(m+2k) \to^*$ INC for some $q \in Q$. (Recall that $k = |Q|$ for the set Q of control states of M.)*

Proof. 'Only if' is obvious.
On any path $p(m) \xrightarrow{\sigma_1} q(m+2k) \xrightarrow{\sigma_2}$ INC we have to cross the level $(m+k)$ when going up as well as when going down to INC (recall that $\ell < k$ for any $r(\ell) \in$ INC). The elementary cycles, which must necessarily appear when going up and down, can be suitably pumped to show the condition in Lemma 17. □

Lemma 19. *Deciding $\sim$-regularity of one-counter processes is in* PTIME.

Proof. We check the condition from Proposition 18. Given $p(m)$, we can compute all $q(m+2k)$ which have finite distances to INC by a polynomial algorithm (recall Point 1 of Lemma 11). When $m = 0$, the reachability of a suitable $q(2k)$ ($q(2k) \to^*$ INC) can be checked straightforwardly. So we can compute all p' such that $p'(0)$ is not $\sim$-regular. Thus $p(m)$ is not $\sim$-regular iff it can reach one of the computed $q(m+2k)$ and $p'(0)$ by positive transitions. The polynomiality follows by the ideas similar to those discussed in the proof of Lemma 10. □

Lemma 20. *Deciding $\sim$-regularity (even) of one-counter nets is* PTIME*-hard.*

Proof. We use a logspace reduction from bisimilarity on finite transition systems which is PTIME-complete [1]. Given a finite transition system $(Q, \mathsf{A}, \{\xrightarrow{a}\}_{a \in \mathsf{A}})$ and $f, g \in Q$, we construct a one counter net which has the following behaviour: in $s_0(m)$, $m > 0$, it has transitions $s_0(m) \xrightarrow{a} s_0(m+1)$, $s_0(m) \xrightarrow{a} s_0(m-1)$, $s_0(m) \xrightarrow{b} f(m)$, $s_0(m) \xrightarrow{b} g(m)$. In $s_0(0)$ we only have $s_0(0) \xrightarrow{a} s_0(1)$ and $s_0(0) \xrightarrow{b} f(0)$. Any state $f(n)$ just mimicks f (not changing the counter); similarly $g(n)$ mimicks g. It is easy to verify that $s_0(n)$ is regular iff $f \sim g$. □

Theorem 21. *Deciding $\sim$-regularity of one-counter processes is* PTIME*-complete.*

Acknowledgements. We thank the anonymous reviewers for useful comments and suggestions.

References

1. J. L. Balcázar, J. Gabarró, and M. Santha. Deciding Bisimilarity is P-Complete. *Formal Asp. Comput.*, 4(6A):638–648, 1992.
2. T. Brázdil, V. Brozek, K. Etessami, A. Kučera, and D. Wojtczak. One-Counter Markov Decision Processes. In *Proc. of SODA*, pages 863–874. IEEE, 2010.
3. S. Demri and A. Sangnier. When Model-Checking Freeze LTL over Counter Machines Becomes Decidable. In *Proc. of FOSSACS*, volume 6014 of *LNCS*, pages 176–190. Springer, 2010.
4. R. v. Glabbeek. The Linear Time – Branching Time Spectrum I; The Semantics of Concrete, Sequential Processes. In J. Bergstra, A. Ponse, and S. Smolka, editors, *Handbook of Process Algebra*, chapter 1, pages 3–99. Elsevier, 2001.

5. S. Göller, R. Mayr, and A. W. To. On the Computational Complexity of Verifying One-Counter Processes. In *Proc. of LICS*, pages 235–244. IEEE Computer Society Press, 2009.
6. C. Haase, S. Kreutzer, J. Ouaknine, and J. Worrell. Reachability in succinct and parametric one-counter automata. In *Proc. of CONCUR*, volume 5710 of *LNCS*, pages 369–383. Springer, 2009.
7. Y. Hirshfeld and M. Jerrum. Bisimulation Equivalence Is Decidable for Normed Process Algebra. In *Proc. of ICALP*, volume 1644 of *LNCS*, pages 412–421. Springer, 1999.
8. D. Janin and I. Walukiewicz. On the Expressive Completeness of the Propositional mu-Calculus with Respect to Monadic Second Order Logic. In *Proc. of CONCUR*, volume 1119 of *LNCS*, pages 263–277. Springer, 1996.
9. P. Jančar. Decidability of Bisimilarity for One-Counter Processes. *Information Computation*, 158(1):1–17, 2000.
10. P. Jančar. Strong Bisimilarity on Basic Parallel Processes is PSPACE-complete. In *Proc. of LICS*, pages 218–227. IEEE Computer Society, 2003.
11. P. C. Kanellakis and S. A. Smolka. CCS Expressions, Finite State Processes, and Three Problems of Equivalence. *Information and Computation*, 86(1):43–68, May 1990.
12. A. Kučera. Efficient Verification Algorithms for One-Counter Processes. In *Proc. of ICALP*, volume 1853 of *LNCS*, pages 317–328. Springer, 2000.
13. A. Kučera and R. Mayr. On the Complexity of Checking Semantic Equivalences between Pushdown Processes and Finite-state Processes. *Inf. Comput.*, 208(7):772–796, 2010.
14. R. Mayr. Process Rewrite Systems. *Information and Computation*, 156(1):264–286, 2000.
15. R. Mayr. Undecidability of Weak Bisimulation Equivalence for 1-Counter Processes. In *Proc. of ICALP*, volume 2719 of *LNCS*, pages 570–583, 2003.
16. R. Milner. *Communication and Concurrency*. International Series in Computer Science. Prentice Hall, 1989.
17. F. Moller and A. M. Rabinovich. Counting on CTL*: on the expressive power of monadic path logic. *Inf. Comput.*, 184(1):147–159, 2003.
18. R. Paige and R. E. Tarjan. Three partition refinement algorithms. *SIAM Journal on Computing*, 16(6):973–989, Dec. 1987.
19. G. Sénizergues. The Bisimulation Problem for Equational Graphs of Finite Out-Degree. *SIAM J. Comput.*, 34(5):1025–1106, 2005.
20. O. Serre. Parity games played on transition graphs of one-counter processes. In *Proc. of FOSSACS*, number 3921 in LNCS. Springer, 2006.
21. J. Srba. Strong Bisimilarity and Regularity of Basic Process Algebra Is PSPACE-Hard. In *Proc. of ICALP*, volume 2380 of *LNCS*, pages 716–727. Springer, 2002.
22. J. Srba. Undecidability of Weak Bisimilarity for PA-Processes. In *Proc. of DLT*, volume 2450 of *LNCS*, pages 197–208. Springer, 2002.
23. J. Srba. *Roadmap of Infinite results*, volume Vol 2: Formal Models and Semantics. World Scientific Publishing Co., 2004. http://www.brics.dk/~srba/roadmap.
24. J. Srba. Beyond Language Equivalence on Visibly Pushdown Automata. *Logical Methods in Computer Science*, 5(1:2), 2009.
25. C. Stirling. Decidability of Bisimulation Equivalence for Pushdown Processes. *unpublished manuscript*, 2000.
26. A. W. To. Model Checking FO(R) over One-Counter Processes and beyond. In *Proc. of CSL*, volume 5771 of *LNCS*, pages 485–499. Springer, 2009.
27. L. G. Valiant and M. Paterson. Deterministic one-counter automata. *J. Comput. Syst. Sci.*, 10(3):340–350, 1975.
28. J. van Benthem. *Modal Correspondence Theory*. PhD thesis, University of Amsterdam, 1976.
29. H.-C. Yen. Complexity Analysis of Some Verification Problems for One-Counter Machines. *unpublished manuscript*, 20xx.