Trace inclusion for one-counter nets revisited

Piotr Hofman [a], Patrick Totzke [b],*

[a] *University of Warsaw, Poland*
[b] *University of Edinburgh, UK*

A B S T R A C T

One-counter nets (OCN) consist of a nondeterministic finite control and a single integer counter that cannot be fully tested for zero. They form a natural subclass of both One-Counter Automata, which allow zero-tests and Petri Nets/VASS, which allow multiple such weak counters. The trace inclusion problem has recently been shown to be undecidable for OCN. In this paper, we contrast the complexity of two natural restrictions which imply decidability.

We show that trace inclusion between a OCN and a *deterministic* OCN is NL-complete, even with arbitrary binary-encoded initial counter-values as part of the input. Secondly, we show that the trace *universality* problem of nondeterministic OCN, which is equivalent to checking trace inclusion between a finite system and a OCN-process, is Ackermann-complete.

© 2017 Elsevier B.V. All rights reserved.

1. Introduction

A fundamental question in formal verification is if the behavior of one process can be reproduced by – or equals that of – another given process. These inclusion and equivalence problems, respectively have been studied for various notions of behavioral preorders and equivalences and for many computational models. Trace inclusion/equivalence asks if the set of *traces*, all emittable sequences of actions, of one process is contained in/equal to that of another. Other than for instance simulation preorder, trace inclusion lacks a strong locality of failures, which makes this problem intractable or even undecidable already for very limited models of computation.

We consider one-counter nets, which consist of a finite control and a single integer counter that cannot be fully tested for zero, in the sense that an empty counter can only restrict possible moves. They are subsumed by One-counter automata (OCA) and thus Pushdown Systems, which allow explicit zero-tests by reading a bottom marker on the stack. At the same time, OCN are a subclass of Petri Nets or Vector Addition Systems with states (VASS): they are exactly the one-dimensional VASS and thus equivalent to Petri Nets with at most one unbounded place.

Related work. Valiant and Paterson [1] show the decidability of the trace equivalence problem for *deterministic* one-counter automata (DOCA). This problem has subsequently been shown to be NL-complete by Böhm, Göller and Jančar [2], assuming fixed initial counter-values. The equivalence of deterministic pushdown automata is known to be decidable [3] and primitive recursive [4], but the exact complexity is still open.

---

* Corresponding author.
    E-mail address: piotrek.hofman@gmail.com (P. Hofman).

Valiant [5] proves the undecidability of both trace inclusion for DOCA and universality for nondeterministic OCA. Jančar, Esparza and Moller [6] consider trace inclusion between Petri Nets and finite systems and prove decidability in both directions. Jančar [7] showed that trace inclusion becomes undecidable if one compares processes of Petri Nets with at least two unbounded places. In [8], the authors show that trace inclusion is undecidable already for (nondeterministic) one-counter nets. Simulation preorder however, is known to be decidable and PSPACE-complete for this model [9–11], which implies a PSPACE upper bound for trace inclusion on DOCN as both relations coincide for deterministic systems.

Higuchi, Tomita and Wakatsuki [12,13] compare the classes of *languages* defined by DOCN with various acceptance modes and in a series of papers consider the respective inclusion problems. They derive procedures that exhaustively search for a bounded witness that work in time and space polynomial in the size of the automata if the initial counter-values are fixed. We show that for monotone relations like trace inclusion or the inclusion of languages defined by acceptance with final states, one can speed up the search for suitable witnesses.

Our contribution. We fix the complexity of two well-known decidable decision problems regarding the traces of one-counter processes. We show that trace inclusion between *deterministic* OCNs is NL-complete. Our upper bound holds even if only the supposedly larger process is deterministic and if (binary encoded) initial counter-values are part of the input. Technically, we show a small witness property for the existence of (possibly long) distinguishing traces. Our certificates are similar to the linear path schemes in [14] and our characterization can be interpreted as showing that the 2-dim. product automaton is flattable in the sense of [15]. The sizes of certificates is polynomial in the number of states of the finite control and they can be verified in space logarithmic in the binary representation of the initial counter-values.

Our second result is that trace universality of *nondeterministic* OCN is Ackermann-complete. This problem is (logspace) inter-reducible with checking trace inclusion between a finite process and a process of a OCN.

2. Background

We write $\mathbb{N}$ for the set of non-negative integers. For any set A, let A^* denote the set of finite strings over A and $\varepsilon \in A^*$ the empty string.

Definition 1. A *one-counter automaton* $\mathcal{A} = (Q, Act, \delta, \delta_0)$ is given by finite sets of control states Q, action labels Act, transitions $\delta \subseteq Q \times Act \times \{-1, 0, 1\} \times Q$ and zero-test transitions $\delta_0 \subseteq Q \times Act \times \{0, 1\} \times Q$. It induces an infinite-state labeled transition system over the state set $Q \times \mathbb{N}$, whose elements will be called *processes* and written as pm where $p \in Q$ and $m \in \mathbb{N}$. The transition relation $\longrightarrow = \longrightarrow_+ \cup \longrightarrow_0$ is partitioned into *positive* and *zero-testing* steps. For states $p, p' \in Q$ and $m, m' \in \mathbb{N}$ these are defined by

1. $pm \xrightarrow{a}_+ p'm' \iff (p, a, (m' - m), p') \in \delta$ and
2. $pm \xrightarrow{a}_0 p'm' \iff (p, a, m', p') \in \delta_0$ and $m = 0$.

Such an automaton is called a *one-counter net* if $\delta_0 = \emptyset$, i.e., if the automaton cannot test if the counter is equal to 0.

When defining a OCN we will omit δ_0 and simply define the net as triple $\mathcal{N} = (Q, Act, \delta)$. Abusing notation, we moreover write $pm \xrightarrow{t} qn$ if a transition $t = (p, a, d, q) \in \delta \cup \delta_0$ justifies a step $pm \xrightarrow{a} qn$.

Definition 2 (*Traces*). Let pm be a process of the OCN $\mathcal{N}$. The *traces* of pm are the elements of the set

$$T_{\mathcal{N}}(pm) = \{a_1 a_2 \ldots a_k \in Act^* \mid \exists qn \; pm \xrightarrow{a_0} \circ \xrightarrow{a_1} \circ \cdots \circ \xrightarrow{a_k} qn\}.$$

We will omit the index $\mathcal{N}$ if is clear from the context. *Trace inclusion* is the decision problem that asks if $T_{\mathcal{A}}(pm) \subseteq T_{\mathcal{B}}(p'm')$ holds for given processes pm and $p'm'$ of nets $\mathcal{A}$ and $\mathcal{B}$, respectively. *Trace universality* asks if $Act^* \subseteq T(pm)$ holds for a given process pm.

An important property of one-counter nets is that the step relation and therefore also trace inclusion is monotone with respect to the counter:

Lemma 1 (*Monotonicity*). *If* $pm \xrightarrow{a} p'm'$ *then* $p(m+1) \xrightarrow{a} p'(m'+1)$. *This in particular means that* $T(pm) \subseteq T(p(m+1))$ *holds for any OCN-process* pm.

Remark 1. In this paper we consider what are sometimes called *realtime* automata, in which no silent (ε-labeled) transitions are present. This is no restriction: the usual syntactic requirement for DPDA, that no state with outgoing ε-transition may have outgoing transitions labeled by $a \neq \varepsilon$, together with the monotonicity of steps in OCN, implies that all states on ε-cycles are essentially deadlocks. One can thus eliminate ε-labeled transitions in logarithmic space.

Normal form. An OCN is *deterministic* if for every $p \in Q$ and $a \in Act$, there is at most one $(p, a, d, q) \in \delta$, at most one $(p, a, d, q) \in \delta_0$ and moreover, if $(p, a, d, p') \in \delta_0$ and $(p, a, d', p'') \in \delta$, then $d' = -1$. It is *complete* if for every $p \in Q$ and $a \in Act$ at least one transition $(p, a, d, q) \in \delta$ exists.

We will w.l.o.g. assume input nets in a certain form, justified by the next lemma. A pair $\mathcal{A}, \mathcal{B}$ of OCNs is in *normal form* if $\mathcal{A}$ is deterministic and $\mathcal{B}$ is complete.

Lemma 2 *(Normal form assumption). Given OCNs $\mathcal{A}$ and $\mathcal{B}$ with state sets N and M, one can construct nets $\mathcal{A}', \mathcal{B}'$ in normal form, with states N and $M' \supseteq M$, respectively, such that for all $(p, n, q, m) \in N \times \mathbb{N} \times M \times \mathbb{N}$*

$$T_{\mathcal{A}}(pm) \subseteq T_{\mathcal{B}}(qn) \iff T_{\mathcal{A}'}(pm) \subseteq T_{\mathcal{B}'}(qn). \tag{1}$$

Moreover, the constructed net $\mathcal{B}'$ is deterministic if the original net $\mathcal{B}$ is.

Proof. Let $\mathcal{A} = (N, Act, \delta_A)$ and $\mathcal{B} = (M, Act', \delta_B)$. If $\mathcal{A}$ is not already deterministic, we can make it so by uniquely re-labeling all its transitions t by actions a_t and adding corresponding transitions (p', a_t, d', q') to the other net $\mathcal{B}$ for any existing $(p', a, d', q') \in \delta_B$, where a is the original label of $t \in \delta_A$. So assume $\mathcal{A}$ is deterministic and pick a new action label $\$ \notin Act$. We add \$-labeled cycles with effect 0 to all states of $\mathcal{A}$: The new net $\mathcal{A}' = (N, Act \cup \{\$\}, \overline{\delta_A})$ has transitions $\overline{\delta_A} = \delta_A \cup \{(s, \$, 0, s) | s \in N\}$. To compensate this, we add \$-cycles to all states of $\mathcal{B}$ in the same way. We add a sink state L (for losing), which has counter-decreasing cycles for all actions, and connect all states without outgoing a-transitions to L by a-labeled transitions. $\mathcal{B}' = (M \cup \{L\}, Act \cup \{\$\}, \overline{\delta_B})$ where

$$\overline{\delta_B} = \delta_B \cup \{(s, \$, 0, s) \mid s \in M\}$$
$$\cup \{(s, a, 0, L) \mid a \in Act \text{ and } s \xrightarrow{a} s' \notin \delta \text{ for any } s' \in M)\}$$
$$\cup \{(L, a, -1, L \mid a \in Act \cup \{\$\}\}.$$

We see that if a word w of length k witnesses non-inclusion $T_{\mathcal{A}}(qn) \nsubseteq T_{\mathcal{B}}(q'n')$ then there is a word that witnesses non-inclusion $T_{\mathcal{A}'}(qn) \nsubseteq T_{\mathcal{B}'}(q'n')$ To see this, observe that in this case, any w-labeled path in $\mathcal{B}'$ that starts in state q' must end in state L. This means any such path takes the initial process $q'n'$ to some process Ln'' where $n'' \leq n' + k$ and now by playing n'' times a label \$ we get a new witness. Conversely, if there is a witness w for $T_{\mathcal{A}'}(qn) \nsubseteq T_{\mathcal{B}'}(q'n')$ then the shortest such witness must be of the form $w = w'\$^k$ where w' does not contain actions \$ because as \$-labeled steps leave any process not in state L unchanged. This means w' witnesses $T_{\mathcal{A}}(qn) \nsubseteq T_{\mathcal{B}}(q'n')$. $\quad \square$

Due to the undecidability of trace inclusion for OCN [8], a direct consequence of Lemma 2 is that $T_{\mathcal{A}}(pm) \subseteq T_{\mathcal{B}}(qn)$ is already undecidable if we allow the net $\mathcal{B}$ to be nondeterministic. Unless otherwise stated, we will from now on assume a DOCN $\mathcal{A} = (Q_{\mathcal{A}}, Act, \delta_{\mathcal{A}})$ and a complete DOCN $\mathcal{B} = (Q_{\mathcal{B}}, Act, \delta_{\mathcal{B}})$.

3. Trace Inclusion for deterministic one-counter nets

We characterize witnesses for non-inclusion $T_{\mathcal{A}}(pm) \nsubseteq T_{\mathcal{B}}(qn)$, starting with some notation to express paths and their effects.

Definition 3 *(OCN paths). Let $\mathcal{N} = (Q, Act, \delta)$ be a OCN and let $t = (p, a, d, p') \in \delta$ be a transition. We write $source(t) = p$, $target(t) = p'$ and $\Delta(t) = d$ for its source and target states and effect, respectively. A path in $\mathcal{N}$ is a sequence $\pi = t_0 t_1 \ldots t_k \in \delta^*$ of transitions where $target(t_i) = source(t_{i+1})$ for every $i < k$. Let $^i\pi$ denote its prefix of length i. The effect $\Delta(\pi)$ and guard $\Gamma(\pi)$ of π are*

$$\Delta(\pi) = \sum_{i=0}^{k} \Delta(t_i) \quad \text{and} \quad \Gamma(\pi) = -\min\{\Delta(^i\pi) \mid 0 \leq i \leq k\}.$$

The path π is *enabled* in process pm (write $pm \xrightarrow{\pi}$) if $\Gamma(\pi) \leq m$. We write $pm \xrightarrow{\pi} p'm'$ if π takes pm to $p'm'$, i.e., if $pm \xrightarrow{\pi}$, $target(\pi) = p'$ and $m' = m + \Delta(\pi)$.

The guard $\Gamma(\pi)$ is the minimal counter-value that is sufficient to traverse the path π while maintaining a non-negative counter-value along the way. This value is always non-negative.

Notice that the absolute values of the effect and guard of a path are bounded by its length. We consider the synchronous product of the control graphs of two given deterministic one-counter nets.

Definition 4 *(Products). The product of nets $\mathcal{A}$ and $\mathcal{B}$ is the finite graph with nodes $V = Q_{\mathcal{A}} \times Q_{\mathcal{B}}$ and $(Act \times \{-1, 0, 1\} \times \{-1, 0, 1\})$-labeled edges E, wheres*

$$(p,q) \xrightarrow{a,d_\mathcal{A},d_\mathcal{B}} (p',q') \in E \text{ iff } p \xrightarrow{a,d_\mathcal{A}} p' \in \delta_\mathcal{A} \text{ and } q \xrightarrow{a,d_\mathcal{B}} q' \in \delta_\mathcal{B}.$$

A *path* in the product is a sequence $\pi = T_0 T_1 \ldots T_k \in E^*$ and defines paths $\pi_\mathcal{A}$ and $\pi_\mathcal{B}$ in nets $\mathcal{A}$ and $\mathcal{B}$, respectively. It is *enabled* in (pm, qn) if $\pi_\mathcal{A}$ and $\pi_\mathcal{B}$ are enabled in pm and qn, respectively. In this case we write $(pm, qn) \xrightarrow{\pi} (p'm', q'n')$ to mean that $pm \xrightarrow{\pi_\mathcal{A}} p'm'$ and $qn \xrightarrow{\pi_\mathcal{B}} q'n'$. We lift the definitions of source and target nodes to paths in the product: $source(\pi) = (source(\pi_A), source(\pi_B)) \in V$, $target(\pi) = (target(\pi_A), target(\pi_B)) \in V$. Moreover, write $\Delta_A(\pi)$, $\Delta_B(\pi)$, $\Gamma_A(\pi)$ and $\Gamma_B(\pi)$ for the effects and guards of π in nets $\mathcal{A}$ and $\mathcal{B}$, respectively.

Since both $\mathcal{A}$ and $\mathcal{B}$ are deterministic and $\mathcal{B}$ is complete, due to our normal form assumption, a trace $w \in T_\mathcal{A}(pm)$ uniquely determines a path from state (p,q) in their product. We therefore identify witnesses for non-inclusion with the paths they induce in the product.

Definition 5 (*Witnesses*). Assume $T_\mathcal{A}(pm) \nsubseteq T_\mathcal{B}(qn)$ for processes pm and qn of $\mathcal{A}$ and $\mathcal{B}$. A *witness* for (pm, qn) is a path π in the product of $\mathcal{A}$ and $\mathcal{B}$ such that $(pm, qn) \xrightarrow{\pi} (p'm', q'n')$ and for some $a \in Act$, $p'm' \xrightarrow{a}$ but $q'n' \xrightarrow{a}$.

Every witness π for (pm, qn) completely exhausts the counter in the process of $\mathcal{B}$: $(pm, qn) \xrightarrow{\pi} (p'm', q'0)$. This is because a process of a complete net can only *not* make an a-step in case the counter is empty.

Example 1. Consider two nets given by self-loops $p \xrightarrow{a,0} p$ and $q \xrightarrow{a,-1} q$, respectively. Their product is the cycle $L = (p,q) \xrightarrow{a,0,-1} (p,q)$ with effects $\Delta_A(L) = 0$ and $\Delta_B(L) = -1$. The only witness for (pm, qn) for initial counter-values $m, n \in \mathbb{N}$ is L^n, which has length polynomial in the sizes of the nets *and* the initial counter-values, but not in the sizes of the nets alone.

The previous example shows that if binary-encoded initial counter-values are part of the input, we can only bound the length of shortest witnesses exponentially. However, we will see that it suffices to consider witnesses of a certain regular form only. This leads to small certificates for non-inclusion, which can be stepwise guessed and verified in space logarithmic in the size of the nets.

A crucial ingredient for our characterization is the monotonicity of witnesses, a direct consequence of the monotonicity of the steps in OCNs stated in Lemma 1:

Lemma 3. *If π is a witness for (pm, qn) then for all $m' \geq m$ and $n' \leq n$ some prefix of π is a witness for (pm', qn').*

The intuition behind the further characterization of witnesses is that in order to show non-inclusion, one looks for a path that is enabled in the process of $\mathcal{A}$ and moreover exhausts the counter in the process of $\mathcal{B}$. Since any sufficiently long path will revisit control states in the product, we can compare such paths with respect to their effect on the counters and see that some are "better" than others. For instance, a cycle that only increments the counter in $\mathcal{B}$ and decrements the one in $\mathcal{A}$ is surely suboptimal considering our goal to find a (shortest) witness. The characterization Theorem 1 essentially states that if a witness exists, then also one that, apart from short paths, combines only the most productive cycles.

Definition 6 (*Loops*). A non-empty path π in the product is called a *cycle* if $source(\pi) = target(\pi)$. Such a cycle is a *loop* if none of its proper subpaths is a cycle. The *slope* of loop π is the ratio $S(\pi) = \Delta_\mathcal{A}(\pi)/\Delta_\mathcal{B}(\pi)$, where for $n > 0$ and $k \in \mathbb{Z}$ we let $n/0 = \infty > k$, $0/0 = 0$ and $-n/0 = -\infty < k$. Based on the effect of a loop we distinguish four types of loops: $(<,<), (>,\geq), (\leq,\geq)$, and $(\geq,<)$. The *type* of π is $Type(\pi) = (\blacktriangleleft, \blacktriangleright)$ iff $\Delta_\mathcal{A}(\pi) \blacktriangleleft 0$ and $\Delta_\mathcal{B}(\pi) \blacktriangleright 0$.

Note that no loop is longer than $|V|$ because it visits exactly one node twice.

Example 2. Consider two DOCN such that their product is the graph depicted below, where we identify transitions with their action labels for simplicity and let $v_0 = (p, p') \in V$. The paths $t_0 t_1 t_2$, $t_3 t_4$ and t_6 are loops with slopes $3/1$, $2/1$ and $1/1$ and types $(>,\geq)$, $(>,\geq)$ and $(<,<)$, respectively.
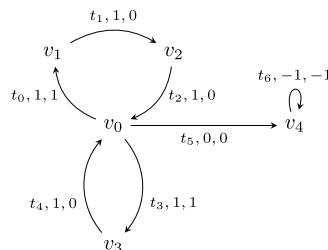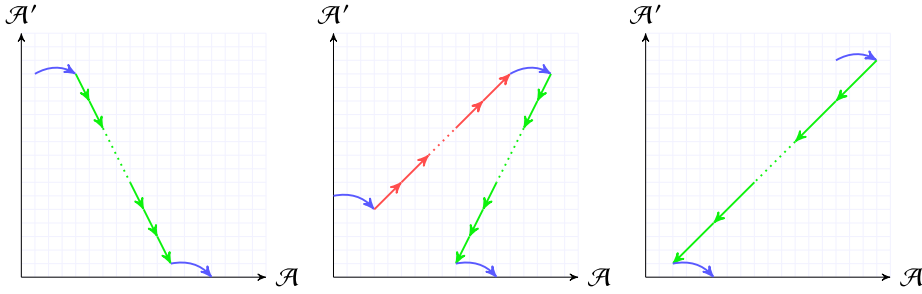
Fig. 1. Illustration of the changes in counter values during paths of forms 1, 2 and 3 as given in Theorem 1. The bent blue arrows indicate the effect of short paths π_i, the straight arrows (in red and green) indicate the counter-effect of loops that are iterated.

The path $(t_0t_1t_2)(t_3t_4)^9 t_5(t_6)^{20}$ is a witness for $(p0, p'10)$ of length 42. By replacing 8 occurrences of the loop (t_3t_4) with $(t_0t_1t_2)^8$ we derive the longer witness $(t_0t_1t_2)^9(t_3t_4)t_5(t_6)^{20}$, which has essentially the same structure but is more efficient in the sense that for the same effect on $\mathcal{B}$ it achieves a higher counter-effect on $\mathcal{A}$.

Theorem 1. *Fix a DOCN $\mathcal{A}$, a complete DOCN $\mathcal{B}$, and let $K \in \mathbb{N}$ be the number of nodes in their product. There is a bound $c \in \mathbb{N}$ that depends polynomially on K, such that the following holds for any two processes pm and qn of $\mathcal{A}$ and $\mathcal{B}$. If $T(pm) \nsubseteq T(qn)$, then there is a witness for (pm, qn) that is either no longer than c or has one of the following forms:*

1. *$\pi_0 L_0^{l_0} \pi_1$, where L_0 is a loop of type $(\geq, <)$ and π_0, π_1 are bounded by c,*
2. *$\pi_0 L_0^{l_0} \pi_1 L_1^{l_1} \pi_2$, where L_0 and L_1 are loops of type $(>, \geq)$ and $(<, <)$ with $S(L_0) > S(L_1)$ and π_0, π_1, π_2 are bounded by c,*
3. *$\pi_0 L_0^{l_0} \pi_1$, where L_0 is a loop of type $(<, <)$ and π_0, π_1 are bounded by c,*

where in all cases, the number of iterations $l_0, l_1 \in \mathbb{N}$ are polynomial in K and the initial counter-values m and n of the given processes. See Fig. 1.

Notice that the bound c in the claim of Theorem 1 depends only on the number of states. Section 4 is dedicated to the proof of this characterization.

The proof of our main result (Theorem 2) will reduce to checking inequalities of binary numbers in a particular form, which are decidable in deterministic logspace.

Proposition 1. *Inequalities of the form $a \cdot b + c \geq a' \cdot b' + c'$ can be verified in $\mathcal{O}(\log(\log(a + b + c + a' + b' + c')))$ deterministic space, i.e., in logspace assuming binary encodings.*

Proof (*sketch*). Checking inequalities of binary numbers, as well as addition and multiplication can be implemented in Boolean circuits of constant depth and polynomial size: these operations are all in *DLOGTIME*-uniform TC^0 [16], a subclass of deterministic logspace. The claim follows from the observation that, as a function class, TC^0 is closed under composition with functions $f : \mathbb{Z} \to \mathbb{Z}$ where $|f(x)| \in O(poly(|x|))$, which is the case for the above operations. A more detailed and direct proof can be found in [17]. □

We now derive a decision procedure for trace inclusion that works in nondeterministic logspace.

Theorem 2. *Let pm and qn be processes of OCN $\mathcal{A}$ and DOCN $\mathcal{B}$, respectively, where m, n are given in binary. There is a nondeterministic algorithm that decides $T(pm) \subseteq T(qn)$ in logarithmic space.*

Proof. Let $\mathcal{A} = (Q_{\mathcal{A}}, Act, \delta_{\mathcal{A}})$ and $\mathcal{B} = (Q_{\mathcal{B}}, Act, \delta_{\mathcal{B}})$, and let $K \in \mathbb{N}$ be the number of states in their product. By Lemma 2, we can assume w.l.o.g. that $\mathcal{A}$ is deterministic and $\mathcal{B}$ is complete and deterministic and so Theorem 1 applies.

If the initial counter-values are $m = n = 0$, Theorem 1 implies a polynomial bound on the length of shortest witnesses. In that case, one can simply stepwise guess and verify a witness, explicitly storing the intermediate processes with binary encoded counter-values in logarithmic space. Such a procedure is impossible with arbitrary initial counter-values as part of the input, because one does not even have the space to memorize them.

For the general case, we argue that one can nondeterministically guess a template (consisting of short paths) and verify in logspace that there is indeed some witness that fits this template. Theorem 1 allows us to either guess a short ($\leq c \in poly(K)$) witness or one of forms 1, 2 or 3, together with matching short paths π_i, L_i. The effect and guard of these paths are bounded by their lengths and hence by c. This means $\mathcal{O}(\log K)$ space suffices to stepwise compute the binary representation of these values and verify that the conditions the form imposes on the types and slopes of the loops are met.

It remains to check if exponents $l_0, l_1 \in \mathbb{N}$ exist, that complete the description of a witness π. We use Lemma 1 to these checks can be implemented in logarithmic space.

For templates of the first two forms, it suffices to check if $m \geq \Gamma_{\mathcal{A}}(\pi_0 L_0)$, because the type of L_0 implies that $\Gamma_{\mathcal{A}}(\pi_0 L_0^l) \leq \Gamma_{\mathcal{A}}(\pi_0 L_0)$ for all $1 < l \in \mathbb{N}$. This means that the process pm of $\mathcal{A}$ can go to, and repeat the loop L_0 arbitrarily often. In case its effect in $\mathcal{B}$ is negative (in templates of form 1), this immediately implies the existence of a suitable l_0. For templates of form 2) the existence of $l_0, l_1 \in \mathbb{N}$ completing the description of a witness is guaranteed because the slope of the first loop is bigger than that of the second.

For templates of the third kind recall that, because $\mathcal{B}$ is complete, a path $\pi = \pi_0 L_0^{l_0} \pi_1$ is a witness iff there is some edge T in the product such that $\Delta_{\mathcal{B}}(T) = -1$ and both $m \geq \Gamma_{\mathcal{A}}(\pi T)$ and $n + \Delta_{\mathcal{B}}(\pi T) = -1$. Equivalently, we can write this as

$$m + \Delta_{\mathcal{A}}(\pi_0 L_0^{l_0}) = m + \Delta_{\mathcal{A}}(\pi_0) + \Delta_{\mathcal{A}}(L_0) \cdot l_0 \geq \Gamma_{\mathcal{A}}(\pi_1 T) \text{ and} \tag{2}$$

$$n + 1 = -\Delta_{\mathcal{B}}(\pi T) = -\Delta_{\mathcal{B}}(\pi_0) - \Delta_{\mathcal{B}}(L_0) \cdot l_0 - \Delta_{\mathcal{B}}(\pi_1 T). \tag{3}$$

Eliminating l_0, we see that this is true iff

$$m + \Delta_{\mathcal{A}}(\pi_0) + \Delta_{\mathcal{A}}(L_0) \cdot \frac{\Delta_{\mathcal{B}}(\pi_0) + \Delta_{\mathcal{B}}(\pi_1) + n}{-\Delta_{\mathcal{B}}(L_0)} \geq \Gamma_{\mathcal{A}}(\pi_1). \tag{4}$$

Simplifying further we can bring this into the form $m \cdot A - n \cdot B \geq C$ where A, B, C are polynomial in c. Now the claim follows from Lemma 1. $\square$

Remark 2. This argument only provides a PSPACE upper bound for the case of one-counter nets with arbitrary binary encoded transition effects. The main reason is that the constant c from Theorem 1, and hence the length of intermediate paths π_i in our path schemes, are bounded by the maximal loop effects. So the size of path schemes, but also the integers we compare in the verification phase become exponential in the size of the input.

4. Proof of Theorem 1

The overall idea of the proof is to rewrite witnesses into one of the canonical forms. More specifically, we introduce a system of path-rewriting rules which simplify witnesses by removing, reducing or changing some loops as in Example 2. We show (in Lemma 4) that these rules preserve witnesses, and in (Lemma 5), that every sequence of successive rule applications must eventually terminate with a normalized path, to which none of the rules is applicable. Such a *reduced* witness can be decomposed as

$$\pi = \pi_0 L_0^{l_0} \pi_1 L_1^{l_1} \ldots \pi_k L_k^{l_k} \pi_{k+1} \tag{5}$$

where k is bounded polynomially in the sizes of nets, the L_i are (pairwise different) loops and the π_i are short, i.e. polynomially bounded in the sizes of the nets. Finally, in Lemma 6, we derive polynomial bounds on the exponents l_i of all but at most two of these loops, which implies the claim Theorem 1.

For the rest of this section let V and E be the sets of nodes and edges in the product of $\mathcal{A}$ and $\mathcal{B}$.

We start with an easy observation: Because no loop L is longer than $|V|$, we conclude that $(\Delta_{\mathcal{A}}(L), \Delta_{\mathcal{B}}(L)) \in \{-V \ldots V\} \times \{-V \ldots V\}$, so there are $F_0 := (2 \cdot |V| + 1)^2$ different values the pair $\Delta_{\mathcal{A}}(L), \Delta_{\mathcal{B}}(L)$ can have. Moreover, if a witness exists, then also one that does not contain different loops with the same effects: if $\pi_0 L_0 \pi_1 L_1 \pi_2$ is a witness where $|\pi_1| > 0$ and L_0, L_1 are two loops with $\Delta(L_0) = \Delta(L_1)$, then either some prefix of $\pi_0 L_0^2 \pi_1 \pi_2$ (if $\Delta_{\mathcal{A}}(L_0) \geq 0$) or some prefix of $\pi_0 \pi_1 L_1^2 \pi_2$ (if $\Delta_A(L_0) < 0$) must also be a witness by Lemma 3. We can therefore consider w.l.o.g. only *sane* paths, which are of the form

$$\pi = \pi_0 L_0^{l_0} \pi_1 L_1^{l_1} \ldots \pi_r L_r^{l_r} \pi_{r+1} \tag{6}$$

where $r \leq F_0$, all π_i are acyclic and all loops have pairwise different effects.

Definition 7 *(Path rewriting rules).* Consider the rules given below.

UUL	**UUR**
$\pi = \pi_0 L_0^{l_0} \pi_1 L_1^{l_1} \pi_2$	$\pi = \pi_0 L_0^{l_0} \pi_1 L_1^{l_1} \pi_2$
$Type(L_0) = (>, \geq)$	$Type(L_0) = (>, \geq)$
$Type(L_1) = (>, \geq)$	$Type(L_1) = (>, \geq)$
$\Delta_{\mathcal{B}}(L_0) \cdot x = \Delta_{\mathcal{B}}(L_1) \cdot y$	$\Delta_{\mathcal{B}}(L_0) \cdot x = \Delta_{\mathcal{B}}(L_1) \cdot y$
$S(L_0) \geq S(L_1)$	$S(L_0) < S(L_1)$
$l_1 - y > 0$	$l_0 - x > |\pi_1 L_1|$
$\rho = \pi_0 L_0^{l_0 + x} \pi_1 L_1^{l_1 - y} \pi_2$	$\rho = \pi_0 L_0^{l_0 - x} \pi_1 L_1^{l_1 + y} \pi_2$

$$
\begin{array}{ccc}
\textbf{UD} & \textbf{DDL} & \textbf{DDR} \\
\pi = \pi_0 L_0^{l_0} \pi_1 L_1^{l_1} \pi_2 & \pi = \pi_0 L_0^{l_0} \pi_1 L_1^{l_1} \pi_2 & \pi = \pi_0 L_0^{l_0} \pi_1 L_1^{l_1} \pi_2 \\
Type(L_0) = (>, \geq) & Type(L_0) = (<, <) & Type(L_0) = (<, <) \\
Type(L_1) = (<, <) & Type(L_1) = (<, <) & Type(L_1) = (<, <) \\
\Delta_{\mathcal{B}}(L_0) \cdot x = -\Delta_{\mathcal{B}}(L_1) \cdot y & \Delta_{\mathcal{B}}(L_0) \cdot x = \Delta_{\mathcal{B}}(L_1) \cdot y & \Delta_{\mathcal{B}}(L_0) \cdot x = \Delta_{\mathcal{B}}(L_1) \cdot y \\
S(L_0) \leq S(L_1) & S(L_0) < S(L_1) & S(L_0) \geq S(L_1) \\
l_0 - x \geq |\pi_1| & l_1 > |L_0| \cdot x + 2|\pi_1| & l_0 - x > 0 \\
l_1 - y > 0 \wedge l_0 - x > 0 & l_1 - y > 0 & \\ \hline
\rho = \pi_0 L_0^{l_0-x} \pi_1 L_1^{l_1-y} \pi_2 & \rho = \pi_0 L_0^{l_0+x} \pi_1 L_1^{l_1-y} \pi_2 & \rho = \pi_0 L_0^{l_0-x} \pi_1 L_1^{l_1+y} \pi_2
\end{array}
$$

Each rule consists of *conditions* (lines above the bar) and a conclusion ρ, which is a path, below the bar. Their names indicate which type of loops are handled: E.g., **UUL** exchanges loops of type $(>, \geq)$ (up) for others of the same type on the left.

We say a rule is *applicable* to a sane path π if there are $0 < x, y, l_0, l_1 \in \mathbb{N}$ and two different loops L_0 and L_1 such that all conditions are satisfied. In this case the rule can rewrite π to ρ, its conclusion and we say ρ is the result of applying the rule to π.

Example 3. Consider Example 2 again: The substitution suggested there is an application of the rule **UUL** to the path $\pi = (t_0 t_1 t_2)(t_3 t_4)^9 t_5 (t_6)^{20}$, where $L_0 = (t_0 t_1 t_2)$, $L_1 = (t_3 t_4)$ and $x = y = 8$. The result is a reduced witness for $(p0, p'10)$ of length 50. Shorter reduced witnesses for $(p0, p'10)$ exist, for example $(t_0 t_1 t_2)^6 t_5 t_6^{16}$, but because of their different loop structure, these cannot be obtained from π by applying rewriting rules, as these do not change the structure, i.e., which loops occur and in which order, of a path. This means that our rules do not necessarily preserve minimality of witnesses.

In the next two Lemmas 4 and 5, we show that the rewriting rules preserve witnesses and that continuous rule application must eventually terminate.

Lemma 4. *If π is a sane witness for $(pm, p'm')$ and ρ is the result of applying one of the rules to π, then ρ is also a sane witness for $(pm, p'm')$.*

Proof. Each rule only modifies the number of times some loops are iterated, and never completely removes a loop. Therefore, sane paths are always rewritten to other sane paths.

Let's say we rewrite $\pi = \pi_0 L_0^{l_0} \pi_1 L_1^{l_1} \pi_2$ to ρ. The key observation is that the conditions of the rule imply that we can always decompose the paths π and ρ into $\pi = \alpha\gamma$ and $\rho = \beta\gamma$, s.t. $\Delta_{\mathcal{B}}(\alpha) = \Delta_{\mathcal{B}}(\beta)$ and $\Delta_{\mathcal{A}}(\alpha) \leq \Delta_{\mathcal{A}}(\beta)$. By monotonicity (Lemma 1) and the assumption that π is a witness, it is therefore sufficient to show that the result ρ is still enabled in the initial position $(pm, p'm')$. We proceed by case distinction for the used rule.

UUL. Since π is a witness, its prefix $\alpha = \pi_0 L_0^{l_0} \pi_1 L_1^{l_1}$ must be enabled in $(pm, p'm')$ and because $Type(L_0) = (>, \geq)$, so is the prefix $\beta = \pi_0 L_0^{l_0+x} \pi_1 L_1^{l_1-y}$ of the result ρ. Assume that $(pm, p'm') \xrightarrow{\alpha} (qn, q'n')$ and $(pm, p'm') \xrightarrow{\beta} (q\hat{n}, q'n')$. The condition $S(L_0) \geq S(L_1)$ of the rule implies that $\hat{n} \geq n \geq \Gamma(\pi_2)$ and therefore that ρ is enabled in $(pm, p'm')$.

UUR. The prefix $\pi_0 L_0^{l_0-x}$ of π must be enabled and since the last condition of the rule demands that $l_0 - x > |\pi_1 L_1|$, so is the path $\pi_0 L_0^{l_0-x} \pi_1 L_1$. The fact that $Type(L_1) = (>, \geq)$, means that also $\pi_0 L_0^{l_0-x} \pi_1 L_1^{l_1+y}$ and therefore the result ρ is enabled in $(pm, p'm')$.

UD. $Type(L_1) = (<, <)$ implies $S(L_1) < \infty$. Since $S(L_0) < S(L_1)$, we know that $S(L_0) < \infty$ and hence $\Delta_{\mathcal{B}}(L_0) > 0$. The path $\pi_0 L_0^{l_0-x}$ is a prefix of π and is therefore enabled in $(pm, p'm')$. As $l_0 - x \geq |\pi_1|$ by assumption, we get that

$$m + \Delta_{\mathcal{A}}(\pi_0 L_0^{l_0-x}) \geq l_0 - x \geq |\pi_1| \geq \Gamma(\pi_1) \tag{7}$$

and similarly, by $\Delta_{\mathcal{B}}(L_0) > 0$,

$$m' + \Delta_{\mathcal{B}}(\pi_0 L_0^{l_0-x}) \geq l_0 - x \geq |\pi_1| \geq \Gamma'(\pi_1). \tag{8}$$

This means that the prefix $\beta = \pi_0 L_0^{l_0-x} \pi_1$ of ρ is enabled in $(pm, p'm')$. Let us now consider the prefix $\alpha = \pi_0 L_0^{l_0-x} L_0^x \pi_1 L_1^y$ of π. Because $\Delta_{\mathcal{B}}(L_0) \cdot x = -\Delta_{\mathcal{B}}(L_1) \cdot y$ we get $\Delta_{\mathcal{B}}(\alpha) = \Delta_{\mathcal{B}}(\beta)$. By $S(L_0) < S(L_1)$ we obtain that $\Delta_{\mathcal{A}}(\alpha) \leq \Delta_{\mathcal{A}}(\beta)$. Because $\pi = \alpha L_1^{l_1-y} \pi_2$ is a witness for $(pm, p'm')$, we can apply Lemma 1 to conclude $\rho = \beta L_1^{l_1-y} \pi_2$ must be a witness for $(pm, p'm')$.

DDL. We know that $m + \Delta_{\mathcal{A}}(\pi_0 L_0^{l_0}) + \Delta_{\mathcal{A}}(\pi_1) \geq \Gamma(L_1^{l_1})$, because π is enabled in $(pm, p'm')$. As L_1 is a type $(<, <)$ loop we also know that $\Delta_{\mathcal{A}}(L_1) < 0$. Therefore, $\Gamma(L_1^{l_1}) \geq l_1$ and

$$m + \Delta_{\mathcal{A}}(\pi_0 L_0^{l_0}) \geq l_1 - \Delta_{\mathcal{A}}(\pi_1). \tag{9}$$

Assume towards a contradiction that $m + \Delta_{\mathcal{A}}(\pi_0 L_0^{l_0}) < \Gamma(L_0^x \pi_1)$. This means that

$$m + \Delta_{\mathcal{A}}(\pi_0 L_0^{l_0}) < \Gamma(L_0^x) + |\pi_1| \leq |L_0| \cdot x + |\pi_1|. \tag{10}$$

This, together with Eq. (9) yields $l_1 - \Delta_{\mathcal{A}}(\pi_1) < |L_0| \cdot x + |\pi_1|$ and thus $l_1 < |L_0| \cdot x + 2|\pi_1|$ which contradicts the condition that $l_1 > |L_0| \cdot x + 2|\pi_1|$. Hence, $m + \Delta_{\mathcal{A}}(\pi_0 L_0^{l_0}) \geq \Gamma(L_0^x \pi_1)$. By the same argument we get that $m' + \Delta_{\mathcal{B}}(\pi_0 L_0^{l_0}) \geq \Gamma'(L_0^x \pi_1)$. So the prefix $\beta = \pi_0 L_0^{l_0+x} \pi_1$ of ρ is enabled in $(pm, p'm')$. Consider the prefix $\alpha = \pi_0 L_0^{l_0} \pi_1 L_1^y$ of π. By the assumption that $\Delta_{\mathcal{B}}(L_0^x) = \Delta_{\mathcal{B}}(L_1^y)$ we get that $\Delta_{\mathcal{B}}(\alpha) = \Delta_{\mathcal{B}}(\beta)$. Because of $S(L_0) < S(L_1)$ we get $\Delta_{\mathcal{A}}(L_0^x) \geq \Delta_{\mathcal{A}}(L_1^y)$ and therefore that $\Delta_{\mathcal{A}}(\alpha) \leq \Delta_{\mathcal{A}}(\beta)$. By Lemma 1 we conclude that the path $\rho = \beta L_1^{l_1-y} \pi_2$ is a witness for $(pm, p'm')$.

DDR. Let $\alpha = \pi_0 L_0^{l_0} \pi_1$ and let $(pm, p'm') \xrightarrow{\alpha} (qn, q'n')$. Due to the type of L_0 and because π is a witness, we know that the prefix $\beta = \pi_0 L_0^{l_0-x} \pi_1 L_1^y$ of ρ is enabled in $(pm, p'm')$. Since $\Delta_{\mathcal{B}}(L_0) \cdot x = \Delta_{\mathcal{B}}(L_1) \cdot y$, we get that $(pm, p'm') \xrightarrow{\beta} (q\hat{n}, q'n')$ for some $\hat{n} \in \mathbb{N}$. The condition $S(L_0) \geq S(L_1)$ of the rule implies that $\Delta_{\mathcal{A}}(L_0^x) \leq \Delta_{\mathcal{A}}(L_1^y) < 0$, and therefore that $\hat{n} \geq n$. We conclude that the path $L_1^{l_1} \pi_2$ is enabled in $(qr, q'r')$ and therefore that $\rho = \pi_0 L_0^{l_0-x} \pi_1 L_1^{l_1+y} \pi_2$ is enabled in $(pm, p'm')$ as required. $\square$

Lemma 5. *Any sequence of successive applications of rules to a given path π must eventually terminate.*

Proof. Consider a path π to which we apply the rewriting rules. W.l.o.g. assume that π is sane, as otherwise no rule is applicable by definition. The *decomposition* of π is the sequence

$$Dec(\pi) = (\pi_0, L_0, l_0)(\pi_1, L_1, l_1) \ldots (\pi_k, L_k, l_k)\pi_{k+1} \tag{11}$$

in $(E^* \times E^* \times \mathbb{N})^* E^*$ such that $\pi = \pi_0 L_0^{l_0} \pi_1 L_1^{l_1} \ldots \pi_k L_k^{l_k} \pi_{k+1}$, where $k \leq F_0$ and for all indices $0 \leq i \leq k$,

1. L_i is a loop,
2. π_i is acyclic,
3. for any two transitions $t \in \pi_i$ and $t' \in L_i$ with $target(t) = target(t')$ holds that $target(L_i) = target(t)$.

The last condition demands that any loop L_i shares exactly one node with the acyclic path π_i it succeeds and thus ensures that the decomposition of a path is unique.

As no application of a rule completely removes all occurrences of loops nor introduces new ones nor touches the intermediate paths, we observe that rule applications only change the exponents l_i in the decomposition of the path.

Based on the order of loops in the decomposition of π, and their potential for rule application, we now define a notion of *weights* for paths, and show that these weights have to strictly decrease along a well-order whenever a rule is applied.

Let $(L_0, L_1, \ldots, L_k)$ be the sequence of loops that occur in the decomposition of π. Let us fix some linear order $\prec$ on $\{L_0, L_1, \ldots, L_k\}$ that satisfies the following conditions for any two different loops L_i, L_j with $i < j$.

1. If $Type(L_i) = Type(L_j) = (>, \geq)$ and $S(L_i) \geq S(L_j)$ then $L_i \prec L_j$.
2. If $Type(L_i) = Type(L_j) = (>, \geq)$ and $S(L_i) < S(L_j)$ then $L_i \succ L_j$.
3. If $Type(L_i) = Type(L_j) = (<, <)$ and $S(L_i) < S(L_j)$ then $L_i \prec L_j$.
4. If $Type(L_i) = Type(L_j) = (<, <)$ and $S(L_i) \geq S(L_j)$ then $L_i \succ L_j$.

Surely, such a linearization exists, as the conditions above only restrict $\prec$ between loops of the same type and slopes are linearly ordered. Consider the permutation $\sigma : \mathbb{N}_{\leq k} \to \mathbb{N}_{\leq k}$ given by $\sigma(i) < \sigma(j) \iff L_i \prec L_j$. The *weight* of π is

$$W(\pi) = (l_{\sigma(k)}, l_{\sigma(k-1)}, \ldots, l_{\sigma(0)}) \in \mathbb{N}^{k+1}. \tag{12}$$

The weight of π is the ordered tuple of exponents l_i of loops that occur in π. Because rules do not change the order of loop occurrences, the path before and after applying a rule have comparable weights. The very definition of weights ensures that rule applications must strictly reduce the weight of a path.

Claim 1. *If ρ is the result of applying a rewriting rule to π then $W(\rho) \sqsubset_{lex} W(\pi)$ where $\sqsubset_{lex}$ is the lexicographic extension of the pointwise ordering of tuples of naturals.*

Assume the decompositions of π, ρ are

$$\begin{aligned} Dec(\pi) &= (\pi_0, L_0, l_0)(\pi_1, L_1, l_1) \ldots (\pi_k, L_k, l_k)\pi_{k+1} \text{ and} \\ Dec(\rho) &= (\pi_0, L_0, l'_0)(\pi_1, L_1, l'_1) \ldots (\pi_k, L_k, l'_k)\pi_{k+1}. \end{aligned} \tag{13}$$

We show for every type of rule that if the occurrences of loop L_i increase then those of some loop L_j with $L_i \prec L_j$ strictly decrease.

If the rule used to derive ρ was **UUL** then $l'_i = l_i + x$ and $l'_j = l_j - y$ for some $i < j$, $0 < x, y$ and type $(>, \geq)$ loops L_i, L_j with $S(L_i) \geq S(L_j)$. By condition 1) in the definition of $\prec$ we get $L_i \prec L_j$.

For rule **UUR** we know $l'_i = l_i - x$ and $l'_j = l_j + y$ for some $0 < x, y$ and type $(>, \geq)$ loops L_i, L_j with $S(L_i) < S(L_j)$. By condition 2) in the definition of $\prec$, we get $L_i \succ L_j$.

For rule **DDL** we know $l'_i = l_i + x$ and $l'_j = l_j - y$ for type $(<, <)$ loops L_i, L_j with $S(L_i) < S(L_j)$. By condition 3) in the definition of $\prec$, we know $L_i \prec L_j$.

For rule **DDR** we know $l'_i = l_i - x$ and $l'_j = l_j + y$ for some $0 < x, y$ and type $(<, <)$ loops L_i, L_j with $S(L_i) > S(L_j)$. So condition 4) in the definition of $\prec$, implies $L_i \succ L_j$.

Lastly, if the rule used to derive ρ was **UD** we immediately see that $l'_i < l_i$ and $l'_j < l_j$, which implies the claim. □

Lemmas 4 and 5 allow us to focus on witnesses that are *reduced*, i.e., which are sane and to which none of the rewriting rules is applicable. We can now derive bounds on the multiplicities of loops in reduced paths.

Lemma 6. *Let $\pi = \pi_0 L_0^{l_0} \pi_1 L_1^{l_1} \pi_2$ be a reduced path where L_0, L_1 are loops occurring with multiplicities $l_0 > 0$ and $l_1 > 0$.*

1. *If $Type(L_0) = Type(L_1) = (>, \geq)$ and $S(L_0) \geq S(L_1)$ then $l_1 \leq |V|$*
2. *If $Type(L_0) = Type(L_1) = (>, \geq)$ and $S(L_0) < S(L_1)$ then $l_0 \leq |\pi_1| + 2|V|$*
3. *If $Type(L_0) = Type(L_1) = (<, <)$ and $S(L_0) < S(L_1)$ then $l_1 < |V|^2 + 2|\pi_1|$*
4. *If $Type(L_0) = Type(L_1) = (<, <)$ and $S(L_0) \geq S(L_1)$ then $l_0 < |V|$*
5. *If $Type(L_0) = (>, \geq), Type(L_1) = (<, <)$ and $S(L_0) \leq S(L_1)$ then $l_0 \leq |\pi_1| + |V|$ or $l_1 \leq |V|$.*

Proof. The fourth condition of any rule is satisfied e.g. by $x = \Delta_{\mathcal{B}}(L_1)$ and $y = \Delta_{\mathcal{B}}(L_0)$. So if $0 < x, y \in \mathbb{N}$ is the smallest satisfying pair we know $x, y \leq |V|$. The bounds are now easily derived by contradiction:

1. *If $l_1 \geq |V|$ then $l_1 - y \geq l_1 - |V| > 0$ and rule **UUL** is applicable.*
2. *If $l_0 > |\pi_1| + 2|V|$ then $l_0 - x > |\pi_1| + 2|V| - x \geq |\pi_1| + |L_1| \geq |\pi_1 L_1|$ and therefore rule **UUR** is applicable.*
3. *If $l_1 \geq |V|^2 + 2|\pi_1|$ then $l_1 \geq |L_0| \cdot x + 2|\pi_1|$ and $l_1 - y \geq l_1 - |V| > 0$, so rule **DDL** is applicable.*
4. *If $l_0 > |V|$ then $l_0 - x > 0$, so rule **DDR** is applicable.*
5. *If $l_1 > |V|$ and $l_0 > |\pi_1| + |V|$, then $l_1 - y > 0$, $l_0 - x > 0$ and $l_0 - x > |\pi_1|$, so rule **UD** is applicable.*

In each case we conclude that one of the rules is applicable, which contradicts the assumption that π is reduced. □

Finally, we are ready to prove Theorem 1.

Theorem 1. *Fix a DOCN $\mathcal{A}$, a complete DOCN $\mathcal{B}$, and let $K \in \mathbb{N}$ be the number of nodes in their product. There is a bound $c \in \mathbb{N}$ that depends polynomially on K, such that the following holds for any two processes pm and qn of $\mathcal{A}$ and $\mathcal{B}$. If $T(pm) \nsubseteq T(qn)$, then there is a witness for (pm, qn) that is either no longer than c or has one of the following forms:*

1. *$\pi_0 L_0^{l_0} \pi_1$, where L_0 is a loop of type $(\geq, <)$ and π_0, π_1 are bounded by c,*
2. *$\pi_0 L_0^{l_0} \pi_1 L_1^{l_1} \pi_2$, where L_0 and L_1 are loops of type $(>, \geq)$ and $(<, <)$ with $S(L_0) > S(L_1)$ and π_0, π_1, π_2 are bounded by c,*
3. *$\pi_0 L_0^{l_0} \pi_1$, where L_0 is a loop of type $(<, <)$ and π_0, π_1 are bounded by c,*

where in all cases, the number of iterations $l_0, l_1 \in \mathbb{N}$ are polynomial in K and the initial counter-values m and n of the given processes.

Proof. We show that we can choose c sufficiently large such that whenever $T(pm) \nsubseteq T(qn)$ but no witness exists that is shorter than c or of forms 1) or 2), then there must be a witness of form 3).

Assume $T(pm) \nsubseteq T(qn)$ and consider a reduced witness π, that is minimal in length: no shorter witness is reduced. Recall that this also means that π is sane: it is of form described in Eq. (6). By monotonicity (Lemma 3) and because π is of minimal length among the reduced witnesses, we see that it cannot contain loops of type $(\leq, \geq)$. Moreover, because π is not of form 1), we can safely assume that it contains only loops of types $(>, \geq)$ and $(<, <)$. This is because if a witness contains two or more different type $(\geq, <)$ loops, then there exists another (sane) witness, that only unfolds the first such loop. If we choose the bound on the length of paths between loops as $F_1 := F_0(2|V| + |V|^2)$, we can write π as

$$\pi = \pi_0 L_0^{l_0} \pi_1 L_1^{l_1} \dots \pi_k L_k^{l_k} \pi_{k+1} \tag{14}$$

where $k \leq F_0$, all $|\pi_i| < F_1$ and the number of iterations of loop L_i is $l_i > |V|$.

Consider a block $\pi_{pos} = L_i^{l_i} \pi_{i+1} L_{i+1}^{l_{i+1}} \pi_{i+2} \dots \pi_j L_j^{l_j}$ that is part of the decomposition above, such that all loops are type $(>, \geq)$. If for indices $i \leq x < y \leq j$ we have $S(L_x) \geq S(L_y)$, then by Lemma 6.1 we get $l_y \leq |V|$. Therefore, π_{pos} can be rewritten to the form

$$\pi_{pos} = L_i^{l_i} \pi_{i+1} L_{i+1}^{l_{i+1}} \pi_{i+2} \dots \pi_j L_j^{l_j} \pi_{j+1} \tag{15}$$

where the lengths of π_i are bounded by $F_2 := F_0 \cdot (|V|^2 + F_1)$ and the slopes of loops are strictly increasing: $S(L_x) < S(L_y)$ for any two indices $i \le x < y \le j$. By Lemma 6.2 this means that $l_x \le |\pi_{x+1}| + 2|V| \le F_2 + 2|V| =: F_3$. We conclude that the prefix $\pi' = L_i^{l_i} \pi_{i+1} L_{i+1}^{l_{i+1}} \pi_{i+2} \ldots \pi_{j-1} L_{j-1}^{l_{j-1}}$ is no longer than $(j - i) \cdot (|V| \cdot F_3 + F_2)$ and therefore

$$\pi_{pos} = \pi' L_j^{l_j} \pi_{j+1} \tag{16}$$

where $|\pi'|$ is bounded by $F_4 := F_0(|V| \cdot F_3 + F_2)$ and $|\pi_{j+1}|$ by F_2.

We continue to show by a similar argument that we can bound the number of iterations of all but the most productive loop in a block consisting of only decreasing (type $(<, <)$) loops. Consider a block $\pi_{neg} = L_i^{l_i} \pi_{i+1} L_{i+1}^{l_{i+1}} \pi_{i+2} \ldots \pi_j L_j^{l_j}$ that is part of the decomposition in Eq. (14), where all loops are type $(<, <)$. If $S(L_x) \ge S(L_y)$ for some indices $i \le x < y \le j$, then by Lemma 6.4 we know $l_y < |V|$. This means that π_{neg} is of the form

$$\pi_{neg} = \pi_i L_i^{l_i} \pi_{i+1} L_{i+1}^{l_{i+1}} \pi_{i+2} \ldots \pi_j L_j^{l_j} \pi_{j+1} \tag{17}$$

where all π_i have lengths bounded by F_2 and $S(L_x) < S(L_y)$ for any two indices $i \le x < y \le j$. By Lemma 6.3 we get $l_y \le |V|^2 + 2|\pi_x| \le |V|^2 + 2F_2 =: F_3'$ and conclude that the suffix $\pi'' = \pi_{i+1} L_{i+1}^{l_{i+1}} \pi_{i+2} \ldots \pi_j L_j^{l_j} \pi_{j+1}$ is no longer than $(j - i) \cdot (|V| \cdot F_3' + F_2)$. Therefore, π_{neg} is of the form

$$\pi_{neg} = \pi_i L_i^{l_i} \pi'' \tag{18}$$

where π_i is bounded by F_2 and π'' by $F_4' := F_0(|V| \cdot F_3' + F_2)$.

Eqs. (16) and (18) characterize the form of maximal subpaths of the witness π in Eq. (14), along which the type of loops does not change. They allow us to write π as

$$\pi = \pi_0 L_0^{l_0} \pi_1 L_1^{l_1} \pi_2 \ldots \pi_k L_k^{l_k} \pi_{k+1} \tag{19}$$

where for all indices $0 \le i < k$:

1. π_i is no longer than $F_5 := F_3 + F_3' + F_4 + F_4'$.
2. All $l_i > |V|$.
3. Consecutive loops L_i and L_{i+1} have different types.
4. If loops L_i, L_j for $0 \le i < j \le k$ have the same type then $S(L_i) < S(L_j)$.

In the remainder of this proof, we further increase the polynomial bound for the gaps π_i between the loops; this allows to conclude that π contains at least one type $(<, <)$ loop and finally, that π is of form 3).

Observe that if all loops L_i in Eq. (19) are of type $(>, \ge)$ then the witness is already of form $\pi = \pi_0 L^l \pi_1$ as in Eq. (16), where π_0, π_1 are short and L is the most effective loop. In this case, consider the run

$$(pm, qn) \xrightarrow{\pi_0 L^l} (p'm', q'n') \tag{20}$$

induced by the prefix $\pi_0 L^l$. Because $\mathcal{B}$ is complete we know $\Delta_{\mathcal{B}}(\pi) = -n$. together with $\Delta_{\mathcal{B}}(\pi_1) \le |\pi_1| \le F_5$ we get $n' \le F_5$. Because $\Gamma_{\mathcal{A}}(\pi_1) \le |\pi_1|$, we know that $l \le |\pi_1| \le F_5$ as otherwise, fewer iterations l would result in a shorter witness and we assumed π to be minimal in length. Hence, we could bound π by $F_6 := F_5 + |V| \cdot F_5 + F_5$. So if we let $c \ge F_6$, our witness π must contain type $(<, <)$ loops as it is assumed not to be no shorter than c.

Finally, fix an index $0 \le x \le k$ such that in Eq. (19), L_x is a loop of type $(<, <)$ with most efficient decrease (minimal slope). That is, π is of form

$$\pi = \pi_0 L_x^{l_x} \pi_1. \tag{21}$$

We now bound both π_0 and π_1 and thereby prove that π is of form 3). We start with the suffix π_1.

If L_x is the only loop of type $(<, <)$, we are done as then $|\pi_1| \le F_5$. Suppose we have two indices $0 \le y < y + 2 \le k$, where both L_y and L_{y+2} are type $(<, <)$. This means that L_{y+1} is of type $(>, \ge)$ with $S(L_{y+1}) < S(L_{y+2})$. By Lemma 6.5 and the fact that $l_{y+2} > |V|$ we know that $l_{y+1} < |\pi_{y+1}| + |V| \le F_6$. So $\pi_{y+1} L_{y+1}^{l_{y+1}} \pi_{y+2}$ is no longer than $2 \cdot F_5 + |V| \cdot F_6 =: F_7$. Applying Lemma 6.3 to L_y and L_{y+2} we get $l_{y+2} \le |V|^2 + 2 \cdot F_7 =: F_8$ and thus $\pi_{y+1} L_{y+1}^{l_{y+1}} \pi_{y+2} L_{y+2}^{l_{y+2}}$ is no longer than $F_9 := F_5 + (|V| \cdot F_6) + F_5 + (|V| \cdot F_8)$. Now the above argument can be repeated for any successive pair of type $(<, <)$ loops in π_1 of which there are at most F_0. So, $|\pi_1| < F_0 \cdot F_9$.

To bound the prefix π_0 in Eq. (21), we recall (point 3 above) that consecutive loops in Eq. (19) have different types and therefore $x \le 1$. In case $x = 0$, we immediately get $|\pi_0| \le F_5$. If $x = 1$, then L_0 is a type $(>, \ge)$ loop with $S(L_0) < S(L_x)$ and so by Lemma 6.5 and point 2), we get $l_0 \le |\pi_1| + |V| < F_6$. This means $|\pi_0| \le 2F_5 + |V| \cdot F_6 = F_7$.

We conclude that $c := F_9 \cdot F_0$ is sufficient to ensure that any witness π, longer than c which is not of form 1) or 2) must have form 3). This completes our argument for the existence of witnesses in the claimed forms.

To see why l_0 and l_1 can always be bounded polynomially in $|V|$ and m' can be seen by looking at the types of the loops involved. For paths of form 1 and 3, L_0 decreases the counter on the right at least once in every iteration. Since the value $m' + \Delta_{\mathcal{B}}(\pi_0)$ before the first iteration is at most $m' + c$, we have $l_0 \leq m' + c$.

Paths of the second form can be decomposed into a prefix $\pi_0 L_0^{l_0}$ and a suffix $\pi_1 L_1^{l_1} \pi_2$, which is a path of form 3. Let $y_0 \in \mathbb{N}$ be minimal such that the effect of the path $\gamma_0 = \pi_0 L_0^0 \pi_1 L_1^{y_0} \pi_2$, in which L_0 is not iterated at all is sufficient to reduce the initial value m' below 0. That is, we have $m' + \Delta_{\mathcal{B}}(\pi_0 L_0^0 \pi_1 L_1^{y_0} \pi_2) \leq 0$. Note that as for forms 1 and 3, we can bound y_0 by $m' + 2c$ and therefore, $|\gamma_0|$ is no larger than $3c + |V| \cdot (m' + 2c)$. This path might not be a witness because it is not enabled on the left side. However, because of the condition on the slopes, there are $x, y \leq |V|$ such that the effect of the loops satisfy

$$\Delta_{\mathcal{B}}(L_0) \cdot x = -\Delta_{\mathcal{B}}(L_1) \cdot y \qquad \text{and} \qquad \Delta_{\mathcal{A}}(L_0) \cdot x > -\Delta_{\mathcal{A}}(L_1) \cdot y. \tag{22}$$

This means, increasing the iterations of the loops L_0 and L_1 by x and y, respectively, does not change the effect of the path on the right, but strictly increases the effect on the left. We increase the iterations $(l_0, l_1) = (0, y_0)$ in γ_0 as suggested above for $\Gamma(\gamma_0) < |\gamma_0| < 3c + |V| \cdot (m' + 2c)$ times. The resulting path $\gamma_1 = \pi_0 L_0^{x_1} \pi_1 L_1^{y_1} \pi_2$ is then surely witness, and iterates the loops not more than $x_1 = 3c + |V| \cdot (m' + 2c)$ and $y_1 = m' + 5c + |V| \cdot (m' + 2c)$ times. $\square$

5. Universality of nondeterministic one-counter nets

To contrast the result of the previous section we now turn to the problem of checking trace inclusion between a finite process and a nondeterministic OCN. This problem is known to be decidable, even for general Petri nets [6] and it can be easily seen to be (logspace) inter-reducible with the trace universality problem, because OCNs are closed under products with finite systems.

For OCN, trace universality can be decided using a simple well-quasi-order based saturation method that determinizes the net on the fly. We will see that this procedure is optimal: The problem is Ackermannian, i.e. it is non-primitive recursive and lies exactly at level ω of the Fast Growing Hierarchy [18].

Let $\mathbb{N}_\perp$ be the set of non-negative integers plus a special least element $\perp$ and let max be the total function that returns the maximal element of any nonempty finite subset and $\perp$ if no such maximal element exists. Consider a set $S \subseteq Q \times \mathbb{N}$ of processes of a OCN $\mathcal{N} = (Q, Act, \delta)$. We lift the definition of traces to sets of processes in the natural way: the *traces* of S are $T(S) = \bigcup_{qn \in S} T(qn)$. By the monotonicity of trace inclusion (Lemma 1), the traces of a finite set of processes are determined only by the traces of its maximal elements.

Definition 8. Let $Q = \{q_1, q_2, \ldots, q_k\}$ be the states-set of some OCN. For a finite set $S \subseteq Q \times \mathbb{N}$ define the *macrostate* as the vector $M_S \in \mathbb{N}_\perp^k$ where for each $0 < i \leq k$, $M_S(i) = M_S(q_i) = \max\{n \mid q_i n \in S\}$. In particular, the macrostate for a singleton set $S = \{q_i n\}$ is the vector with value n at the i-th coordinate and $\perp$ on all others. The *norm* of a macrostate $M \in \mathbb{N}_\perp^k$ is $|M|_\infty = \max\{M(i) \mid 0 < i \leq k\}$. We define a step relation $\overset{a}{\Longrightarrow}$ for all $a \in Act$ on the set of macrostates as follows:

$$(n_1, n_2, \ldots, n_k) \overset{a}{\Longrightarrow} (m_1, m_2, \ldots, m_k) \tag{23}$$

iff $m_i = \max\{n \mid \exists n_j \neq \perp. q_j n_j \overset{a}{\longrightarrow} q_i n\}$ for all $0 < i \leq k$. The *traces* of macrostate M are $T(M) = \bigcup_{0 < i \leq k} T(q_i M(i))$, where $T(q \perp) = \emptyset$. For two macrostates M, N we say M is *covered* by N and write $M \sqsubseteq N$, if it is pointwise smaller, i.e., $M(i) \leq N(i)$ for all $0 < i \leq k$. For convenience, we will write $\{q_1 = n_1, q_2 = n_2, \ldots, q_l = n_l\}$ for the macrostate with value $M(i) = n_i$ whenever $q_i = n_i$ is listed and $\perp$ otherwise.

Steps on macrostates correspond to the classical powerset construction and each macrostate represents the finite set of possible processes the OCN can be in, where all non-maximal ones (w.r.t. their counter-value) are pruned out.

The next lemma directly follows from these definitions and monotonicity (Lemma 1).

Lemma 7.

1. *The covering-order $\sqsubseteq$ is a well-quasi-order on $\mathbb{N}_\perp^k$, the set of all macro-states. Moreover, $M \sqsubseteq N$ implies $T(M) \subseteq T(N)$.*
2. *If $M \overset{a}{\Longrightarrow} N$ then $|N|_\infty \leq |M|_\infty + 1$.*
3. *For any finite set $S \subseteq Q \times \mathbb{N}$ it holds that $T(S) = T(M_S)$.*

Dealing with macrostates allows us to treat universality as a reachability problem: By point 3 of Lemma 7 we see that a process qn is *not* trace universal, $Act^* \neq T(qn)$, if and only if $M_{\{qn\}} \Longrightarrow^* (\perp, \perp, \ldots, \perp)$. We take the perspective of a pathfinder, whose goal it is to reach $(\perp)^k$.

We can decide universality by stepwise guessing a shortest terminating path from the initial macrostate, and thus a witness for non-universality. Whenever we see a macrostate that covers one of its predecessors, we can safely discard this candidate, because omitting the intermediate path would result in a shorter witness by Lemma 7.1.

We show non-primitive recursiveness by reduction from the control state reachability problem for incrementing counter machines [19,18].

Definition 9 *(Counter machines).* A (Minsky)-counter machine (CM) is an automaton with finitely many states Q, finitely many *counters* $C_1, \ldots, C_k$, and transitions are of the form $Q \times Act \times Q$ where Act is $\{inc, dec, ifz\} \times \{1, 2, \ldots, k\}$. A *configuration* of such a CM consists of a state and a valuation of the counters. Performing a transition $(p, (op, i), q)$ changes a configuration precisely: the state changes from p to q and we make operation op on the counter c_i, where inc, dec and ifz mean increment, decrement and zero-test, respectively. Such a step is forbidden if the requested operation is dec and the value of c_i is 0, or if $c_i > 0$ and the operation is ifz.

An *incrementing counter machine* (ICM) is a CM in which counters can spontaneously increment without performing any transitions. Such increments we call incrementing errors. *Control state reachability* is the decision problem that asks if there is a run of a given CM from an initial configuration to some given state $q_f \in Q$.

Our reduction is based on the following simple observation. Consider a OCN $\mathcal{N} = (Q, Act, \delta)$ that contains a *universal* state U: it has self-loops $U \xrightarrow{a,0} U \in \delta$ for every action $a \in Act$. A Pathfinder who wants to prove non-universality must avoid macrostates with $M(U) \neq \bot$, because no continuation of a path leading to such a macrostate can be a witness. We can use this idea to construct macrostates that prevent Pathfinder from making certain actions.

Definition 10 *(Obstacles).* Let $S \subseteq Act$ be a set of actions in a OCN that contains a universal state U. A state $q \in Q$ is called an *S-obstacle* if $q \xrightarrow{a,0} U \in \delta$ for all actions $a \in S$. We say q *ignores* S, if $q \xrightarrow{a,0} q \in \delta$ for all $a \in S$.

Note that if a macrostate contains an *S*-obstacle, then Pathfinder must avoid all actions of *S*. In order to remove an obstacle, Pathfinder must play an action that is not the label of any of its incoming transitions.

Theorem 3. *Trace universality for OCN is not primitive recursive.*

Proof. By reduction from the control state reachability problem for ICM, which has non-primitive recursive complexity [19]. We construct a OCN-process *Init*(0) that is not universal iff a given ICM reaches a final state from its initial configuration. The idea is to enforce a faithful simulation of the ICM by pathfinder, who wants to show non-universality of the OCN by stepwise rewriting the initial macrostate $\{Init = 0\}$ to the all-bottom-macrostate $\bot^l$.

We construct a OCN $\mathcal{N}$ which has a unique action for every transition of the ICM, as well as actions τ_i that indicate incrementing errors for every counter c_i, and actions $\sharp$ and \$ to mark the beginning and end of a run, respectively. This way we make sure there is a strict correspondence between words and ICM-runs. The states of $\mathcal{N}$ are

- a new initial state *Init* and a universal state *U*,
- a state q_i for every state q_i of the ICM,
- a state C_i for every counter c_i of the ICM,
- a state *Z*, which ignores every action but the end marker \$. State *Z* will be used to access the constant 0.

A configuration $q(c_1, c_2, \ldots, c_k)$ of the ICM is represented by a macrostate $\{q = 0, Z = 0, C_1 = c_1, C_2 = c_2, \ldots, C_k = c_k\}$. We will define the transitions of $\mathcal{N}$ such that the only way for Pathfinder to reach $\bot^l$ is by rewriting the initial macrostate $\{Init = 0\}$ to the one representing the initial ICM configuration and then to stepwise announce the transitions of an accepting run of the ICM. Using the idea of obstacles, we define the rules of the net $\mathcal{N}$ so that the only way Pathfinder can avoid the universal state U and reach the macrostate $\bot^k$ is by first transforming the initial macrostate $\{Init = 0\}$ to the one that represents the initial ICM configuration and then announcing transitions (as well as actions demanding increment errors) of a valid and accepting run of the ICM.

Initialization. To set up $M_0 = \{q_0 = 0, Z = 0, C_0 = 0, C_1 = 0, \ldots, C_k = 0\}$, representing the initial ICM configuration, we add $\sharp$-labeled transitions with effect 0 from *Init* to q_0, Z and C_i for all $0 \leq i \leq k$. Moreover, we make *Init* an obstacle for every action but $\sharp$. This way, Pathfinder has to play $\sharp$ as the first move (and set up M_0) in order to avoid a universal macrostate.

Finite control. For any transition $t = q \xrightarrow{(a,i)} q'$ of the ICM, we add a transition $q \xrightarrow{t,0} q'$ to $\mathcal{N}$ that, in a macrostate-step, will replace the value 0 in dimension q by $\bot$ and introduce value 0 in dimension q'. Moreover, we make every state q an obstacle for all actions announcing ICM-transitions not originating in q. This prevents Pathfinder from announcing transitions from q unless the current macrostate has $M(q) = 0$ and $M(q_i) = \bot$ for all $q_i \neq q$.

Simulation of the counters. Every transition operates on one of the counters c_i for $0 \leq i \leq k$. Below we list the corresponding transitions in the OCN $\mathcal{N}$ for this counter. Every state of $\mathcal{N}$ not explicitly mentioned ignores the action in question. In the macrostate, the values of these states are therefore unchanged.

increments For ICM-transitions t that increase the ith counter, $\mathcal{N}$ contains a t-labeled transition from state C_i to C_i with effect $+1$. Additionally, to deal with spontaneous increment errors, there is a τ_i-labeled increasing self-loop in state C_i.

decrements For ICM-transitions t that decrease the ith counter, $\mathcal{N}$ contains a t-labeled transition from state C_i to C_i with effect -1.

This means that the next macrostate M could lose the value for this counter and have $M(C_i) = \bot$ if previously, the value was 0. In that case, the decrementing step from value 0 to value 0 is valid in the ICM because it can first (silently) increment and then do the (visible) decrement step. In order to avoid losing the state C_i in the macrostate, the OCN contains a transition $Z \xrightarrow{t,0} C_i$ from the constant-zero state Z to state C_i. Recall that Z is present in the macrostate because Z ignores every action except end marker \$. Consequently, no correctly set up macrostate will set $M(C_i) = \bot$.

zero-tests For ICM-transitions t that test the ith counter for 0, we add a t-labeled transition $C_i \xrightarrow{t,-1} U$ from state C_i to the universal state. This prevents Pathfinder from using these actions if the current macrostate has $M(C_i) > 0$ because it would make the next macrostate universal. If however $M(C_i) = 0$, such a step is safe because the punishing transition is not enabled in the OCN-process $C_i 0$.

Lastly, we only add transitions to $\mathcal{N}$ so that the final state q_f is the only original ICM-state which is not an obstacle for \$. This prevents Pathfinder from playing the end-marker \$ unless the simulation has reached the final state. □

In order to estimate an upper bound, we recall a recent result of Figueira, Figueira, Schmitz, Schnoebelen [18], that allows us to provide the exact complexity of the OCN trace universality problem in terms of its level in the Fast-Growing Hierarchy. The main idea is to estimate the maximal length of a path in the well quasi order based algorithm using bounds on the difference of sizes of consecutive configurations.

Theorem 4. *Trace universality of OCN is Ackermannian.*

Definition 11 (*Fast-growing hierarchy*). Consider the family of functions $F_n : \mathbb{N} \to \mathbb{N}$ where for $x, k \in \mathbb{N}$,

$$F_0(x) = x + 1 \quad \text{and} \qquad F_{k+1}(x) = F_k^{x+1}(x).$$

Here, F^k denotes the k-fold application of F. Moreover, define $F_\omega(x) = F_x(x)$ for the first limit ordinal ω. For $k \leq \omega$, $\mathfrak{F}_k$ denotes the least class of functions that contains all constants and is closed under substitution, sum, projections, limited recursion and applications of functions F_n for $n \leq k$.

Already $\mathfrak{F}_2$ contains all elementary functions and the union $\bigcup_{k \in \mathbb{N}} \mathfrak{F}_k$ of all finite levels contains exactly the primitive-recursive functions. A function is called *Ackermannian* if it is in $\mathfrak{F}_\omega \setminus \bigcup_{k \in \mathbb{N}} \mathfrak{F}_k$.

A sequence $x_0, x_1, \ldots, x_l$ of macrostates is called *good* if there are indices $0 \leq i < j \leq l$ such that $x_i \sqsubseteq x_j$ and *bad* otherwise. The sequence is *t-controlled* by $f : \mathbb{N} \to \mathbb{N}$ if $|x_i|_\infty < f(i + t)$ for every index $0 \leq i \leq l$.

Theorem 5 ([18]). *Let $f : \mathbb{N} \to \mathbb{N}$ be a monotone function in $\mathfrak{F}_\gamma$ such that $f(x) \geq \max\{1, x\}$ for some $\gamma \geq 1$. There is a function $L_{k,f}(t)$ in $\mathfrak{F}_{k+\gamma-1}$ that computes a bound on the maximal length of bad sequences in $\mathbb{N}_\bot^k$ that are t-controlled by f.*

Corollary 1. *Trace universality of OCN is Ackermannian.*

Proof. By Theorem 3, it suffices to show that the problem is in $\mathfrak{F}_\omega$. Recall the procedure that, for a given process pm of a net with k control states, guesses a shortest terminating path from the initial macrostate (a witness for non-universality), and stops unsuccessfully if a macrostate covers one that has been seen before. The time and space requirements of this procedure are bounded in terms of the longest non-increasing (w.r.t. covering) sequence of k-dimensional macrostates. These are bad sequences where the norm of the initial macrostate is m, the counter-value of the process to check for universality. By point 2 of Lemma 7, such sequences are m-controlled by the successor function $f(x) = x + 1$, which is in $\mathfrak{F}_1$. By Theorem 5, computing the bound and running the procedure above is in $\mathfrak{F}_k$. As k is part of the input, this yields a procedure in $\mathfrak{F}_\omega$. □

6. Conclusion

We have shown NL-completeness of the trace inclusion problem for deterministic one-counter nets, where initial counter-values are part of the input. Our proof is based on a characterization of the shape of possible witnesses in terms of a small number of polynomially-sized templates. Realizability of such templates can be verified in space logarithmic only in the size of the underlying state space. Our procedure is therefore independent of the number of action symbols and transitions in the input nets. To prove the characterization theorem we use witness rewriting rules, the correctness of which crucially

depends on the monotonicity of trace inclusion w.r.t. counter-values. In fact, we only make use of this property in the net on the left but similarly one can define rules that exploit only the monotonicity in the process on the right. With some additional effort one can extend this argument also for trace inclusion between DOCN and DOCA or vice versa (see [17]).

The second part of the paper explores the complexity of the universality problem for nondeterministic OCN, and trace inclusion between finite systems and OCN that easily reduces to OCN universality. Here we show that the simplest known algorithm which uses a well-quasi-order based saturation technique has already optimal complexity: The problem is Ackermannian, i.e., not primitive recursive.

Acknowledgements

References

[1] L. Valiant, M.S. Paterson, Deterministic one-counter automata, J. Comput. System Sci. 10 (3) (1975) 340–350, http://dx.doi.org/10.1016/S0022-0000(75)80005-5.
[2] S. Böhm, S. Göller, P. Jančar, Equivalence of deterministic one-counter automata is nl-complete, in: STOC, 2013, pp. 131–140.
[3] G. Sénizergues, L(a) = l(b)? Electron. Notes Theor. Comput. Sci. 9 (1997) 43.
[4] C. Stirling, Deciding dpda equivalence is primitive recursive, in: ICALP, 2002, pp. 821–832.
[5] L. Valiant, Decision Procedures for Families of Deterministic Pushdown Automata, Ph.D. thesis, University of Warwick, July 1973.
[6] P. Jančar, J. Esparza, F. Moller, Petri nets and regular processes, J. Comput. System Sci. 59 (3) (1999) 476–503.
[7] P. Jančar, Undecidability of bisimilarity for Petri nets and some related problems, Theoret. Comput. Sci. 148 (2) (1995) 281–301.
[8] P. Hofman, R. Mayr, P. Totzke, Decidability of weak simulation on one-counter nets, in: LICS, 2013, pp. 203–212.
[9] A. Abdulla, K. Čerāns, Simulation is decidable for one-counter nets, in: CONCUR, 1998, pp. 253–268.
[10] P. Jančar, A. Kučera, F. Moller, Simulation and bisimulation over one-counter processes, in: STACS, 2000, pp. 334–345.
[11] P. Hofman, S. Lasota, R. Mayr, P. Totzke, Simulation over one-counter nets is PSPACE-complete, in: FSTTCS, 2013, pp. 515–526.
[12] K. Higuchi, E. Tomita, M. Wakatsuki, A polynomial-time algorithm for checking the inclusion for strict deterministic restricted one-counter automata, IEICE Trans. 78-D (4) (1995) 305–313.
[13] K. Higuchi, M. Wakatsuki, E. Tomita, A polynomial-time algorithm for checking the inclusion for real-time deterministic restricted one-counter automata which accept by final state, IEICE Trans. 78-D (8) (1995) 939–950.
[14] M. Blondin, A. Finkel, S. Göller, C. Haase, P. McKenzie, Reachability in two-dimensional vector addition systems with states is PSPACE-complete, in: Proceedings of the 30th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS'15), ACM Press, Kyoto, Japan, 2015, pp. 32–43, http://arxiv.org/abs/1412.4259.
[15] J. Leroux, G. Sutre, Flat counter automata almost everywhere!, in: D. Peled, Y.-K. Tsay (Eds.), Automated Technology for Verification and Analysis, in: Lecture Notes in Computer Science, vol. 3707, Springer, Berlin/Heidelberg, 2005, pp. 489–503.
[16] W. Hesse, E. Allender, D.A.M. Barrington, Uniform constant-depth threshold circuits for division and iterated multiplication, J. Comput. System Sci. 65 (4) (2002) 695–716, http://dx.doi.org/10.1016/S0022-0000(02)00025-9, Special Issue on Complexity 2001, http://www.sciencedirect.com/science/article/pii/S0022000002000259.
[17] P. Totzke, Inclusion Problems for One-Counter Systems, Ph.D. thesis, LFCS, University of Edinburgh, May 2014.
[18] D. Figueira, S. Figueira, S. Schmitz, P. Schnoebelen, Ackermannian and primitive-recursive bounds with Dickson's lemma, in: LICS, 2011, pp. 269–278.
[19] S. Demri, R. Lazić, LTL with the freeze quantifier and register automata, ACM Trans. Comput. Log. 10 (3) (2009) 16:1–16:30, http://dx.doi.org/10.1145/1507244.1507246.