HAL
open science

Branching-time model checking of one-counter processes

Stefan Göller, Markus Lohrey

To cite this version:

Stefan Göller, Markus Lohrey. Branching-time model checking of one-counter processes. 27th International Symposium on Theoretical Aspects of Computer Science - STACS 2010, Inria Nancy Grand Est & Loria, Mar 2010, Nancy, France. pp.405-416. inria-00455367

HAL Id: inria-00455367

https://inria.hal.science/inria-00455367v1

Submitted on 10 Feb 2010

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Symposium on Theoretical Aspects of Computer Science 2010 (Nancy, France), pp. 405-416
www.stacs-conf.org

BRANCHING-TIME MODEL CHECKING OF ONE-COUNTER PROCESSES

STEFAN GÖLLER [1] AND MARKUS LOHREY [2]

[1] Universität Bremen, Fachbereich Mathematik und Informatik
E-mail address: goeller@informatik.uni-bremen.de

[2] Universität Leipzig, Institut für Informatik
E-mail address: lohrey@informatik.uni-leipzig.de

ABSTRACT. One-counter processes (OCPs) are pushdown processes which operate only on a unary stack alphabet. We study the computational complexity of model checking computation tree logic (CTL) over OCPs. A PSPACE upper bound is inherited from the modal μ-calculus for this problem. First, we analyze the periodic behaviour of CTL over OCPs and derive a model checking algorithm whose running time is exponential only in the number of control locations and a syntactic notion of the formula that we call leftward until depth. Thus, model checking fixed OCPs against CTL formulas with a fixed leftward until depth is in P. This generalizes a result of the first author, Mayr, and To for the expression complexity of CTL's fragment EF. Second, we prove that already over some fixed OCP, CTL model checking is PSPACE-hard. Third, we show that there already exists a fixed CTL formula for which model checking of OCPs is PSPACE-hard. For the latter, we employ two results from complexity theory: (i) Converting a natural number in Chinese remainder presentation into binary presentation is in logspace-uniform NC^1 and (ii) PSPACE is AC^0-serializable. We demonstrate that our approach can be used to answer further open questions.

1. Introduction

Pushdown automata (PDAs) (or recursive state machines) are a natural model for sequential programs with recursive procedure calls, and their verification problems have been studied extensively. The complexity of model checking problems for PDAs is quite well understood: The reachability problem for PDAs can be solved in polynomial time [4, 10]. Model checking modal μ-calculus over PDAs was shown to be EXPTIME-complete in [29], and the global version of the model checking problem has been considered in [7, 21, 22]. The EXPTIME lower bound for model checking PDAs also holds for the simpler logic CTL and its fragment EG [28], even for a fixed formula (data complexity) [5] or a fixed PDA (expression complexity). On the other hand, model checking PDAs against the logic EF (another natural fragment of CTL) is PSPACE-complete [28], and again the lower bound still holds if either the formula or the PDA is fixed [4]. Model checking

1998 ACM Subject Classification: F.4.1; F.1.3.
Key words and phrases: model checking, computation tree logic, complexity theory.
The second author would like to acknowledge the support by DFG research project GELO.

SYMPOSIUM ON THEORETICAL ASPECTS OF COMPUTER SCIENCE

© S. Göller and M. Lohrey
Creative Commons Attribution-NoDerivs License

problems for various fragments and extensions of PDL (Propositional Dynamic Logic) over PDAs were studied in [12].

One-counter processes (OCPs) are Minsky counter machines with just one counter. They can also be seen as a special case of PDAs with just one stack symbol, plus a non-removable bottom symbol which indicates an empty stack (and thus allows to test the counter for zero) and hence constitute a natural and fundamental computational model. In recent years, model checking problems for OCPs received increasing attention [13, 15, 23, 25]. Clearly, all upper complexity bounds carry over from PDAs. The question, whether these upper bounds can be matched by lower bounds was just recently solved for several important logics: Model checking modal μ-calculus over OCPs is PSPACE-complete. The PSPACE upper bound was shown in [23], and a matching lower bound can easily be shown by a reduction from emptiness of alternating unary finite automata, which was shown to be PSPACE-complete in [18, 19]. This lower bound even holds if either the OCP or the formula is fixed. The situation becomes different for the fragment EF. In [13], it was shown that model checking EF over OCPs is in the complexity class P^{NP} (the class of all problems that can be solved on a deterministic polynomial time machine with access to an oracle from NP). Moreover, if the input formula is represented succinctly as a directed acyclic graph, then model checking EF over OCPs is also hard for P^{NP}. For the standard (and less succinct) tree representation for formulas, only hardness for the class $\mathsf{P}^{\mathsf{NP}[\log]}$ (the class of all problems that can be solved on a deterministic polynomial time machine which is allowed to make $O(\log(n))$ many queries to an oracle from NP) was shown in [13]. In fact, there already exists a fixed EF formula such that model checking this formula over a given OCP is hard for $\mathsf{P}^{\mathsf{NP}[\log]}$, i.e., the data complexity is $\mathsf{P}^{\mathsf{NP}[\log]}$-hard.

In this paper we consider the model checking problem for CTL over OCPs. By the known upper bound for the modal μ-calculus [23] this problem belongs to PSPACE. First, we analyze the combinatorics of CTL model checking over OCPs. More precisely, we analyze the periodic behaviour of the set of natural numbers that satisfy a given CTL formula in a given control location of the OCP (Thm. 4.1). By making use of Thm. 4.1, we can derive a model checking algorithm whose running time is exponential only in the number of control locations and a syntactic measure on CTL formulas that we call leftward until depth (Thm. 4.2). As a corollary, we obtain that model checking a fixed OCP against CTL formulas of fixed leftward until depth lies in P. This generalizes a recent result from [13], where it was shown that the expression complexity of EF over OCPs lies in P. Next, we focus on lower bounds. We show that model checking CTL over OCPs is PSPACE-complete, even if we fix either the OCP (Thm. 5.3) or the CTL formula (Thm. 7.2). The proof of Thm. 5.3 uses a reduction from QBF. We have to construct a fixed OCP for which we can construct for a given unary encoded number i CTL formulas that express, when interpreted over our fixed OCP, whether the current counter value is divisible by 2^i and whether the ith bit in the binary representation of the current counter value is 1, respectively. For the proof of Thm. 7.2 (PSPACE-hardness of data complexity for CTL) we use two techniques from complexity theory, which to our knowledge have not been applied in the context of verification so far: (i) the existence of small depth circuits for converting a number from Chinese remainder representation to binary representation and (ii) the fact that PSPACE-computations are serializable in a certain sense (see Sec. 6 for details). One of the main obstructions in getting lower bounds for OCPs is the fact that OCPs are well suited for testing divisibility properties of the counter value and hence can deal with numbers in Chinese remainder representation, but it is not clear how to deal with numbers in binary representation. Small depth circuits for converting a number from Chinese remainder representation to binary representation are the key in order to overcome this obstruction.

We are confident that our new lower bound techniques described above can be used for proving further lower bounds for OCPs. We present two other applications of our techniques in Sec. 8:

(i) We show that model checking EF over OCPs is complete for P^{NP} even if the input formula is represented by a tree (Thm. 8.1) and thereby solve an open problem from [13]. (ii) We improve a lower bound on a decision problem for one-counter Markov decision processes from [6] (Thm. 8.2). The following table summarizes the picture on the complexity of model checking for PDAs and OCPs. Our new results are marked with (*).

Logic	**PDA**	**OCP**
modal μ-calculus	EXPTIME-complete	PSPACE-complete
modal μ-calculus, fixed formula	EXPTIME-complete	PSPACE-complete
modal μ-calculus, fixed system	EXPTIME-complete	PSPACE-complete
CTL, fixed formula	EXPTIME-complete	PSPACE-complete (*)
CTL, fixed system	EXPTIME-complete	PSPACE-complete (*)
CTL, fixed system, fixed leftward until depth	EXPTIME-complete	in P (*)
EF	PSPACE-complete	P^{NP}-complete (*)
EF, fixed formula	PSPACE-complete	$\mathsf{P}^{\mathsf{NP}[\log]}$-hard, in P^{NP}
EF, fixed system	PSPACE-complete	in P

Missing proofs due to space restrictions can be found in the full version of this paper [14].

2. Preliminaries

We denote the naturals by $\mathbb{N} = \{0, 1, 2, \ldots\}$. For $i, j \in \mathbb{N}$ let $[i, j] = \{k \in \mathbb{N} \mid i \leq k \leq j\}$ and $[j] = [1, j]$. In particular $[0] = \emptyset$. For $n \in \mathbb{N}$ and $i \geq 1$, let $\mathrm{bit}_i(n)$ denote the ith least significant bit of the binary representation of n, i.e., $n = \sum_{i \geq 1} 2^{i-1} \cdot \mathrm{bit}_i(n)$. For every finite and non-empty subset $M \subseteq \mathbb{N} \setminus \{0\}$, define $\mathrm{LCM}(M)$ to be the *least common multiple* of all numbers in M. It is known that $2^k \leq \mathrm{LCM}([k]) \leq 4^k$ for all $k \geq 9$ [20]. As usual, for a possibly infinite alphabet A, A^* (resp. A^ω) denotes the set of all finite (resp. infinite) words over A. Let $A^\infty = A^* \cup A^\omega$ and $A^+ = A^* \setminus \{\varepsilon\}$, where ε is the empty word. The length of a finite word w is denoted by $|w|$. For a word $w = a_1 a_2 \cdots a_n \in A^*$ (resp. $w = a_1 a_2 \cdots \in A^\omega$) with $a_i \in A$ and $i \in [n]$ (resp. $i \geq 1$), we denote by w_i the ith letter a_i. A nondeterministic finite automaton (NFA) is a tuple $A = (S, \Sigma, \delta, s_0, S_f)$, where S is a finite set of *states*, Σ is a *finite alphabet*, $\delta \subseteq S \times \Sigma \times S$ is the *transition relation*, $s_0 \in S$ is the *initial state*, and $S_f \subseteq S$ is a set of *final states*. We assume some basic knowledge in complexity theory, see e.g. [1] for more details.

3. One-counter processes and computation tree logic

Fix a countable set $\mathcal{P}$ of *propositions*. A *transition system* is a triple $T = (S, \{S_p \mid p \in \mathcal{P}\}, \rightarrow)$, where S is the set of *states*, $\rightarrow \subseteq S \times S$ is the set of *transitions* and $S_p \subseteq S$ for all $p \in \mathcal{P}$ with $S_p = \emptyset$ for all but finitely many $p \in \mathcal{P}$. We write $s_1 \rightarrow s_2$ instead of $(s_1, s_2) \in \rightarrow$. The set of all *finite* (resp. *infinite*) *paths* in T is $\mathrm{path}_+(T) = \{\pi \in S^+ \mid \forall i \in [|\pi| - 1] : \pi_i \rightarrow \pi_{i+1}\}$ (resp. $\mathrm{path}_\omega(T) = \{\pi \in S^\omega \mid \forall i \geq 1 : \pi_i \rightarrow \pi_{i+1}\}$). For a subset $U \subseteq S$ of states, a (finite or infinite) path π is called a *U-path* if $\pi \in U^\infty$.

A *one-counter process* (OCP) is a tuple $\mathbb{O} = (Q, \{Q_p \mid p \in \mathcal{P}\}, \delta_0, \delta_{>0})$, where Q is a finite set of *control locations*, $Q_p \subseteq Q$ for all $p \in \mathcal{P}$ with $Q_p = \emptyset$ for all but finitely many $p \in \mathcal{P}$, $\delta_0 \subseteq Q \times \{0, 1\} \times Q$ is a set of *zero transitions*, and $\delta_{>0} \subseteq Q \times \{-1, 0, 1\} \times Q$ is a set of *positive transitions*. The *size* of the OCP $\mathbb{O}$ is $|\mathbb{O}| = |Q| + \sum_{p \in \mathcal{P}} |Q_p| + |\delta_0| + |\delta_{>0}|$. The transition system defined by $\mathbb{O}$ is $T(\mathbb{O}) = (Q \times \mathbb{N}, \{Q_p \times \mathbb{N} \mid p \in \mathcal{P}\}, \rightarrow)$, where $(q, n) \rightarrow (q', n + k)$ if and only

if either $n = 0$ and $(q, k, q') \in \delta_0$, or $n > 0$ and $(q, k, q') \in \delta_{>0}$. A *one-counter net* (OCN) is an OCP, where $\delta_0 \subseteq \delta_{>0}$. For $(q, k, q') \in \delta_0 \cup \delta_{>0}$ we usually write $q \xrightarrow{k} q'$.

More details on the temporal logic CTL can be found for instance in [2]. *Formulas* φ of CTL are defined by the following grammar, where $p \in \mathcal{P}$:

$$\varphi \quad ::= \quad p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \exists \mathsf{X}\varphi \mid \exists \varphi \mathsf{U} \varphi \mid \exists \varphi \mathsf{WU} \varphi.$$

Given a transition system $T = (S, \{S_p \mid p \in \mathcal{P}\}, \rightarrow)$ and a CTL formula φ, we define the semantics $[\![\varphi]\!]_T \subseteq S$ by induction on the structure of φ as follows: $[\![p]\!]_T = S_p$ for each $p \in \mathcal{P}$, $[\![\neg\varphi]\!]_T = S \setminus [\![\varphi]\!]_T$, $[\![\varphi_1 \wedge \varphi_2]\!]_T = [\![\varphi_1]\!]_T \cap [\![\varphi_2]\!]_T$, $[\![\exists \mathsf{X}\varphi]\!]_T = \{s \in S \mid \exists s' \in [\![\varphi]\!]_T : s \rightarrow s'\}$, $[\![\exists \varphi_1 \mathsf{U} \varphi_2]\!]_T = \{s \in S \mid \exists \pi \in \mathrm{path}_+(T) : \pi_1 = s, \pi_{|\pi|} \in [\![\varphi_2]\!]_T, \forall i \in [|\pi| - 1] : \pi_i \in [\![\varphi_1]\!]_T\}$, $[\![\exists \varphi_1 \mathsf{WU} \varphi_2]\!]_T = [\![\exists \varphi_1 \mathsf{U} \varphi_2]\!]_T \cup \{s \in S \mid \exists \pi \in \mathrm{path}_\omega(T) : \pi_1 = s, \forall i \geq 1 : \pi_i \in [\![\varphi_1]\!]_T\}$. We also write $(T, s) \models \varphi$ (or briefly $s \models \varphi$ if T is clear from the context) for $s \in [\![\varphi]\!]_T$. We introduce the usual abbreviations $\varphi_1 \vee \varphi_2 = \neg(\neg\varphi_1 \wedge \neg\varphi_2)$, $\forall \mathsf{X}\varphi = \neg\exists \mathsf{X}\neg\varphi$, $\exists \mathsf{F}\varphi = \exists(p \vee \neg p)\mathsf{U}\varphi$, and $\exists \mathsf{G}\varphi = \exists \varphi \mathsf{WU}(p \wedge \neg p)$ for some $p \in \mathcal{P}$. Formulas of the CTL-fragment EF are given by the following grammar, where $p \in \mathcal{P}$: $\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \exists \mathsf{X}\varphi \mid \exists \mathsf{F}\varphi$. The *size* of CTL formulas is defined as follows: $|p| = 1$, $|\neg\varphi| = |\exists \mathsf{X}\varphi| = |\varphi| + 1$, $|\varphi_1 \wedge \varphi_2| = |\varphi_1| + |\varphi_2| + 1$, $|\exists \varphi_1 \mathsf{U} \varphi_2| = |\exists \varphi_1 \mathsf{WU} \varphi_2| = |\varphi_1| + |\varphi_2| + 1$.

4. CTL on OCPs: Periodic behaviour and upper bounds

The goal of this section is to prove a periodicity property of CTL over OCPs, which implies an upper bound for CTL on OCPs, see Thm. 4.2. As a corollary, we state that for a fixed OCP, CTL model checking restricted to formulas of fixed leftward until depth (see the definition below) can be done in polynomial time. We define the *leftward until depth* lud of CTL formulas inductively as follows: $\mathrm{lud}(p) = 0$ for $p \in \mathcal{P}$, $\mathrm{lud}(\neg\varphi) = \mathrm{lud}(\exists \mathsf{X}\varphi) = \mathrm{lud}(\varphi)$, $\mathrm{lud}(\varphi_1 \wedge \varphi_2) = \max\{\mathrm{lud}(\varphi_1), \mathrm{lud}(\varphi_2)\}$, $\mathrm{lud}(\exists \varphi_1 \mathsf{U} \varphi_2) = \mathrm{lud}(\exists \varphi_1 \mathsf{WU} \varphi_2) = \max\{\mathrm{lud}(\varphi_1) + 1, \mathrm{lud}(\varphi_2)\}$. A similar definition of until depth can be found in [24], but there the until depth of $\exists \varphi_1 \mathsf{U} \varphi_2$ is 1 plus the maximum of the until depths of φ_1 and φ_2. Note that $\mathrm{lud}(\varphi) \leq 1$ for every EF formula φ.

Let us fix an OCP $\mathbb{O} = (Q, \{Q_p \mid p \in \mathcal{P}\}, \delta_0, \delta_{>0})$ for the rest of this section. Let $|Q| = k$ and define $K = \mathrm{LCM}([k])$ and $K_\varphi = K^{\mathrm{lud}(\varphi)}$ for each CTL formula φ.

Theorem 4.1. *For all* CTL *formulas* φ*, all* $q \in Q$ *and all* $n, n' > 2 \cdot |\varphi| \cdot k^2 \cdot K_\varphi$ *with* $n \equiv n'$ *mod* K_φ*:*

$$(q, n) \in [\![\varphi]\!]_{T(\mathbb{O})} \iff (q, n') \in [\![\varphi]\!]_{T(\mathbb{O})}. \tag{4.1}$$

Proof sketch. We prove the theorem by induction on the structure of φ. We only treat the difficult case $\varphi = \exists \psi_1 \mathsf{U} \psi_2$ here. Let $T = \max\{2 \cdot |\psi_i| \cdot k^2 \cdot K_{\psi_i} \mid i \in \{1, 2\}\}$. Let us prove equivalence (4.1). Note that $K_\varphi = \mathrm{LCM}\{K \cdot K_{\psi_1}, K_{\psi_2}\}$ by definition. Let us fix an arbitrary control location $q \in Q$ and naturals $n, n' \in \mathbb{N}$ such that $2 \cdot |\varphi| \cdot k^2 \cdot K_\varphi < n < n'$ and $n \equiv n'$ mod K_φ. We have to prove that $(q, n) \in [\![\varphi]\!]_{T(\mathbb{O})}$ if and only if $(q, n') \in [\![\varphi]\!]_{T(\mathbb{O})}$. For this, let $d = n' - n$, which is a multiple of K_φ. We only treat the "if"-direction here and recommend the reader to consult [14] for helpful illustrations. So let us assume that $(q, n') \in [\![\varphi]\!]_{T(\mathbb{O})}$. To prove that $(q, n) \in [\![\varphi]\!]_{T(\mathbb{O})}$, we will use the following claim.

Claim: Assume some $[\![\psi_1]\!]_{T(\mathbb{O})}$-path $\pi = [(q_1, n_1) \rightarrow (q_2, n_2) \rightarrow \cdots \rightarrow (q_l, n_l)]$ with $n_i > T$ for all $i \in [l]$ and $n_1 - n_l \geq k^2 \cdot K \cdot K_{\psi_1}$. Then there exists a $[\![\psi_1]\!]_{T(\mathbb{O})}$-path from (q_1, n_1) to $(q_l, n_l + K \cdot K_{\psi_1})$, whose counter values are all strictly above $T + K \cdot K_{\psi_1}$.

The claim tells us that paths that lose height at least $k^2 \cdot K \cdot K_{\psi_1}$ and whose states all have counter values strictly above T can be flattened (without changing the starting state) by height $K \cdot K_{\psi_1}$.

Proof of the claim. For each counter value $h \in \{n_i \mid i \in [l]\}$ that appears in π, let $\mu(h) = \min\{i \in [l] \mid n_i = h\}$ denote the minimal position in π whose corresponding state has counter value h. Define $\Delta = k \cdot K_{\psi_1}$. We will be interested in $k \cdot K$ many consecutive intervals (of counter values) each of size Δ. Define the bottom $b = n_1 - (k \cdot K) \cdot \Delta$. Formally, an *interval* is a set $I_i = [b + (i-1) \cdot \Delta, b + i \cdot \Delta]$ for some $i \in [k \cdot K]$. Since each interval has size $\Delta = k \cdot K_{\psi_1}$, we can think of each interval I_i to consist of k consecutive *sub-intervals* of size K_{ψ_1} each. Note that each sub-interval has two extremal elements, namely its *upper* and *lower boundary*. Thus all k sub-intervals have $k+1$ boundaries in total. Hence, by the pigeonhole principle, for each interval I_i, there exists some $c_i \in [k]$ and two distinct boundaries $\beta(i,1) > \beta(i,2)$ of distance $c_i \cdot K_{\psi_1}$ such that the control location of π's earliest state of counter value $\beta(i,1)$ agrees with the control location of π's earliest state of counter value $\beta(i,2)$, i.e., formally $q_{\mu(\beta(i,1))} = q_{\mu(\beta(i,2))}$. Observe that flattening the path π by gluing together π's states at position $\mu(\beta(i,1))$ and $\mu(\beta(i,2))$ (for this, we add $c_i \cdot K_{\psi_1}$ to each counter value at a position $\geq \beta(i,2)$) still results in a $[\![\psi_1]\!]_{T(\mathbb{O})}$-path by induction hypothesis, since we reduced the height of π by a multiple of K_{ψ_1}. Our overall goal is to flatten π by gluing together states only of certain intervals such that we obtain a path whose height is in total by precisely $K \cdot K_{\psi_1}$ smaller than π's. Recall that there are $k \cdot K$ many intervals. By the pigeonhole principle there is some $c \in [k]$ such that $c_i = c$ for at least K many intervals I_i. By gluing together $\frac{K}{c} \in \mathbb{N}$ pairs of states of distance $c \cdot K_{\psi_1}$ each, we reduce π's height by exactly $\frac{K}{c} \cdot c \cdot K_{\psi_1} = K \cdot K_{\psi_1}$. This proves the claim.

Let us finish the proof the "if"-direction. Since by assumption $(q, n') \in [\![\varphi]\!]_{T(\mathbb{O})}$, there exists a finite path $\pi = (q_1, n_1) \to (q_2, n_2) \to \cdots \to (q_l, n_l)$, where $\pi[1, l-1]$ is a $[\![\psi_1]\!]_{T(\mathbb{O})}$-path, $(q, n') = (q_1, n_1)$, and where $(q_l, n_l) \in [\![\psi_2]\!]_{T(\mathbb{O})}$. To prove $(q, n) \in [\![\varphi]\!]_{T(\mathbb{O})}$, we will assume that $n_j > T$ for each $j \in [l]$. The case when $n_j = T$ for some $j \in [l]$ can be proven similarly. Assume first that the path $\pi[1, l-1]$ contains two states whose counter difference is at least $k^2 \cdot K \cdot K_{\psi_1} + K_\varphi$ which is (strictly) greater than $k^2 \cdot K \cdot K_{\psi_1}$. Since K_φ is a multiple of $K \cdot K_{\psi_1}$ by definition, we can apply the above claim $\frac{K_\varphi}{K \cdot K_{\psi_1}} \in \mathbb{N}$ many times to $\pi[1, l-1]$. This reduces the height by K_φ. We repeat this flattening process of $\pi[1, l-1]$ by height K_φ as long as possible, i.e., until any two states have counter difference smaller than $k^2 \cdot K \cdot K_{\psi_1} + K_\varphi$. Let σ denote the $[\![\psi_1]\!]_{T(\mathbb{O})}$-path starting in (q, n') that we obtain from $\pi[1, l-1]$ by this process. Thus, σ ends in some state, whose counter value is congruent n_{l-1} modulo K_φ (since we flattened $\pi[1, l-1]$ by a multiple of K_φ). Since K_φ is in turn a multiple of K_{ψ_2}, we can build a path σ' which extends the path σ by a single transition to some state that satisfies ψ_2 by induction hypothesis. Moreover, by our flattening process, the counter difference between any two states in σ' is at most $k^2 \cdot K \cdot K_{\psi_1} + K_\varphi \leq 2 \cdot k^2 \cdot K_\varphi$. Recall that $T = \max\{2 \cdot |\psi_i| \cdot k^2 \cdot K_{\psi_i} \mid i \in \{1, 2\}\}$. As

$$n \;>\; 2 \cdot |\varphi| \cdot k^2 \cdot K_\varphi \;=\; 2 \cdot (|\varphi| - 1 + 1) \cdot k^2 \cdot K_\varphi \;\geq\; T + 2 \cdot k^2 \cdot K_\varphi,$$

it follows that the path that results from σ' by subtracting d from each counter value (this path starts in (q, n)) is strictly above T. Moreover, since d is a multiple of K_{ψ_1} and K_{ψ_2}, this path witnesses $(q, n) \in [\![\varphi]\!]_{T(\mathbb{O})}$ by induction hypothesis. ∎

The following result can be obtained basically by using the standard model checking algorithm for CTL on finite systems (see e.g. [2]) in combination with Thm. 4.1.

Theorem 4.2. *For a given one-counter process $\mathbb{O} = (Q, \{Q_p \mid p \in \mathcal{P}\}, \delta_0, \delta_{>0})$, a CTL formula φ, a control location $q \in Q$, and $n \in \mathbb{N}$ given in binary, one can decide $(q, n) \in [\![\varphi]\!]_{T(\mathbb{O})}$ in time $O(\log(n) + |Q|^3 \cdot |\varphi|^2 \cdot 4^{|Q| \cdot \mathrm{lud}(\varphi)} \cdot |\delta_0 \cup \delta_{>0}|)$.*

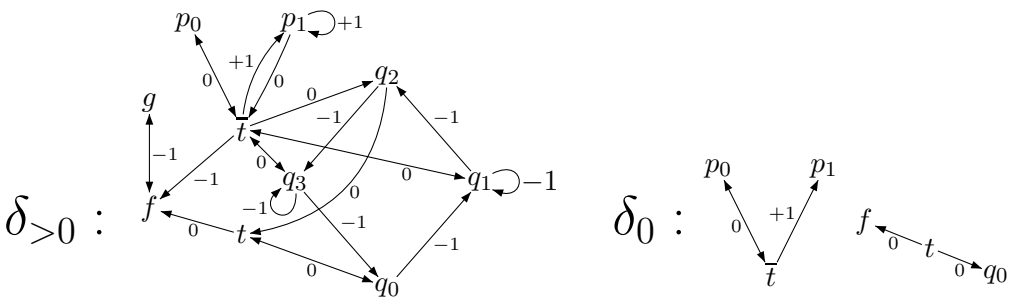

Figure 1: The one-counter net $\mathbb{O}$ for which CTL model checking is PSPACE-hard

As a corollary, we can deduce that for every fixed OCP $\mathbb{O}$ and every fixed k the question if for a given state s and a given CTL formula φ with $\mathrm{lud}(\varphi) \leq k$, we have $(T(\mathbb{O}), s) \models \varphi$, is in P. This generalizes a result from [13], stating that the expression complexity of EF over OCPs is in P.

5. Expression complexity for CTL is hard for PSPACE

The goal of this section is to prove that model checking CTL is PSPACE-hard already over a fixed OCN. We show this via a reduction from the well-known PSPACE-complete problem QBF. Our lower bound proof is separated into three steps. In step one, we define a family of CTL formulas $(\varphi_i)_{i\geq 1}$ such that over the fixed OCN $\mathbb{O}$ that is depicted in Fig. 1 we can express (non-)divisibility by 2^i. In step two, we define a family of CTL formulas $(\psi_i)_{i\geq 1}$ such that over $\mathbb{O}$ we can express if the ith bit in the binary representation of a natural is set to 1. In our final step, we give the reduction from QBF. For step one, we need the following simple fact which characterizes divisibility by powers of two (recall that $[n] = \{1, \ldots, n\}$, in particular $[0] = \emptyset$):

$$\forall n \geq 0, i \geq 1 : \ 2^i \text{ divides } n \ \Leftrightarrow \ (2^{i-1} \text{ divides } n \ \wedge \ |\{n' \in [n] \mid 2^{i-1} \text{ divides } n'\}| \text{ is even}) \quad (5.1)$$

The set of propositions of $\mathbb{O}$ in Fig. 1 coincides with its control locations. Recall that $\mathbb{O}$'s zero transitions are denoted by δ_0 and $\mathbb{O}$'s positive transitions are denoted by $\delta_{>0}$. Since $\delta_0 \subseteq \delta_{>0}$, $\mathbb{O}$ is indeed an OCN. Note that both t and $\overline{t}$ are control locations of $\mathbb{O}$. Now we define a family of CTL formulas $(\varphi_i)_{i\geq 1}$ such that for each $n \in \mathbb{N}$ we have: (i) $(t, n) \models \varphi_i$ if and only if 2^i divides n and (ii) $(\overline{t}, n) \models \varphi_i$ if and only if 2^i does *not* divide n. On first sight, it might seem superfluous to let the control location t represent divisibility by powers of two and the control location $\overline{t}$ to represent non-divisibility by powers of two since CTL allows negation. However the fact that we have *only one* family of formulas $(\varphi_i)_{i\geq 1}$ to express both divisibility and non-divisibility is a crucial technical subtlety that is necessary in order to avoid an exponential blowup in formula size. By making use of (5.1), we construct the formulas φ_i inductively. First, let us define the auxiliary formulas test $= t \vee \overline{t}$ and $\varphi_\diamond = q_0 \vee q_1 \vee q_2 \vee q_3$. Think of $\varphi_\diamond$ to hold in those control locations that altogether are situated in the "diamond" in Fig. 1. We define

$$\begin{aligned} \varphi_1 &= \text{test} \wedge \exists\mathsf{X}\ (f \wedge \mathsf{EF}(f \wedge \neg\exists\mathsf{X} g)) \text{ and} \\ \varphi_i &= \text{test} \ \wedge \ \exists\mathsf{X}\ \big(\exists(\varphi_\diamond \wedge \exists\mathsf{X}\varphi_{i-1})\ \mathsf{U}\ (q_0 \wedge \neg\exists\mathsf{X} q_1)\big) \text{ for } i > 1. \end{aligned}$$

Since φ_{i-1} is only used once in φ_i, we get $|\varphi_i| \in O(i)$. The following lemma states the correctness of the construction.

Lemma 5.1. *Let $n \geq 0$ and $i \geq 1$. Then*

- $(t, n) \models \varphi_i$ *if and only if* 2^i *divides* n.

- $(\overline{t}, n) \models \varphi_i$ *if and only if* 2^i *does not divide* n.

Proof sketch. The lemma is proved by induction on i. The induction base for $i = 1$ is easy to check. For $i > 1$, observe that φ_i can only be true either in control location t or $\overline{t}$. Note that the formula right to the until symbol in φ_i expresses that we are in q_0 and that the current counter value is zero. Also note that the formula left to the until symbol requires that $\varphi_\diamond$ holds, i.e., we are always in one of the four "diamond control locations". In other words, we decrement the counter by moving along the diamond control locations (by possibly looping at q_1 and q_3) and always check if $\exists\mathsf{X}\varphi_{i-1}$ holds, just until we are in q_0 and the counter value is zero. Since there are transitions from q_1 and q_3 to $\overline{t}$ (but not to t), the induction hypothesis implies that the formula $\exists\mathsf{X}\varphi_{i-1}$ can be only true in q_1 and q_3 as long as the current counter value is not divisible by 2^{i-1}. Similarly, since there are transitions from q_0 and q_2 to t (but not to $\overline{t}$), the induction hypothesis implies that the formula $\exists\mathsf{X}\varphi_{i-1}$ can be only true in q_0 and q_2 if the current counter value is divisible by 2^{i-1}. With (5.1) this implies the lemma. ∎

For expressing if the ith bit of a natural is set to 1, we make use of the following simple fact:

$$\forall n \geq 0, i \geq 1 : \text{bit}_i(n) = 1 \iff |\{n' \in [n] \mid 2^{i-1} \text{ divides } n'\}| \text{ is odd} \tag{5.2}$$

Let us now define a family of CTL formulas $(\psi_i)_{i\geq 1}$ such that for each $n \in \mathbb{N}$ we have $\text{bit}_i(n) = 1$ if and only if $(\overline{t}, n) \models \psi_i$. We set $\psi_1 = \varphi_1$ and $\psi_i = \overline{t} \wedge \exists\mathsf{X}\,((q_1 \vee q_2) \wedge \mu_i)$, where $\mu_i = \exists(\varphi_\diamond \wedge \exists\mathsf{X}\varphi_{i-1})\,\mathsf{U}\,(q_0 \wedge \neg\exists\mathsf{X}q_1)$ for each $i > 1$. Due to the construction of ψ_i and since $|\varphi_i| \in O(i)$, we obtain that $|\psi_i| \in O(i)$. The following lemma states the correctness of the construction.

Lemma 5.2. *Let* $n \geq 0$ *and let* $i \geq 1$. *Then* $(\overline{t}, n) \models \psi_i$ *if and only if* $\text{bit}_i(n) = 1$.

Let us sketch the final step of the reduction from QBF. For this, let us assume some quantified Boolean formula $\alpha = Q_k x_k\, Q_{k-1} x_{k-1} \cdots Q_1 x_1 : \beta(x_1, \ldots, x_k)$, where β is a Boolean formula over variables $\{x_1, \ldots, x_k\}$ and $Q_i \in \{\exists, \forall\}$ is a quantifier for each $i \in [k]$. Think of each truth assignment $\vartheta : \{x_1, \ldots, x_k\} \to \{0, 1\}$ to correspond to the natural number $n(\vartheta) \in [0, 2^k - 1]$, where $\text{bit}_i(n(\vartheta)) = 1$ if and only if $\vartheta(x_i) = 1$, for each $i \in [k]$. Let $\widehat{\beta}$ be the CTL formula that is obtained from β by replacing each occurrence of x_i by ψ_i, which corresponds to applying Lemma 5.2. It remains to describe how we deal with quantification. Think of this as to consecutively incrementing the counter from state $(\overline{t}, 0)$ as follows. First, setting the variable x_k to 1 will correspond to adding 2^{k-1} to the counter and getting to state $(\overline{t}, 2^{k-1})$. Setting x_k to 0 on the other hand will correspond to adding 0 to the counter and hence remaining in state $(\overline{t}, 0)$. Next, setting x_{k-1} to 1 corresponds to adding to the current counter value 2^{k-2}, whereas setting x_{k-1} to 0 corresponds to adding 0, as expected. These incrementation steps can be achieved using the formulas φ_i from Lemma 5.1. Finally, after setting variable x_1 either to 0 or 1, we verify if the CTL formula $\widehat{\beta}$ holds. Formally, let $\bigcirc_i = \wedge$ if $Q_i = \exists$ and $\bigcirc_i = \rightarrow$ if $Q_i = \forall$ for each $i \in [k]$ (recall that $Q_k, \ldots, Q_1$ are the quantifiers of our quantified Boolean formula α). Let $\theta_1 = Q_1\mathsf{X}\,((p_0 \vee p_1) \bigcirc_1 \exists\mathsf{X}\,\widehat{\beta})$ and for $i \in [2, k]$:

$$\theta_i = Q_i\mathsf{X}\,\left((p_0 \vee p_1) \bigcirc_i \exists\left((p_0 \vee \exists\mathsf{X}\,(\overline{t} \wedge \varphi_{i-1}))\,\mathsf{U}\,(\overline{t} \wedge \neg\varphi_{i-1} \wedge \theta_{i-1})\right)\right).$$

Then, it can be show that α is valid if and only if $(\overline{t}, 0) \in [\![\theta_k]\!]_{T(\mathbb{O})}$.

Theorem 5.3. CTL *model checking of the fixed OCN* $\mathbb{O}$ *from Fig. 1 is* PSPACE*-hard.*

Note that the constructed CTL formula has leftward until depth that depends on the size of α. By Thm. 4.2 this cannot be avoided unless P = PSPACE. Observe that in order to express divisibility by powers of two, our CTL formulas $(\varphi_i)_{i\geq 0}$ have linearly growing leftward until depth.

6. Tools from complexity theory

For Sec. 7 and 8 we need some concepts from complexity theory. By $\mathsf{P}^{\mathsf{NP}[\log]}$ we denote the class of all problems that can be solved on a polynomially time bounded deterministic Turing machines which can have access to an NP-oracle only logarithmically many times, and by P^{NP} the corresponding class without the restriction to logarithmically many queries. Let us briefly recall the definition of the circuit complexity class NC^1, more details can be found in [26]. We consider Boolean circuits $C = C(x_1, \ldots, x_n)$ built up from AND- and OR-gates. Each input gate is labeled with a variable x_i or a negated variable $\neg x_i$. The output gates are linearly ordered. Such a circuit computes a function $f_C : \{0,1\}^n \to \{0,1\}^m$, where m is the number of output gates, in the obvious way. The *fan-in of a circuit* is the maximal number of incoming wires of a gate in the circuit. The *depth of a circuit* is the number of gates along a longest path from an input gate to an output gate. A *logspace-uniform* NC^1*-circuit family* is a sequence $(C_n)_{n\geq 1}$ of Boolean circuits such that for some polynomial $p(n)$ and constant c: (i) C_n contains at most $p(n)$ many gates, (ii) the depth of C_n is at most $c \cdot \log(n)$, (iii) the fan-in of C_n is at most 2, (iv) for each m there is at most one circuit in $(C_n)_{n\geq 1}$ with exactly m input gates, and (v) there exists a logspace transducer that computes on input 1^n a representation (e.g. as a node-labeled graph) of the circuit C_n. Such a circuit family computes a partial mapping on $\{0,1\}^*$ in the obvious way (note that we do not require to have for every $n \geq 0$ a circuit with exactly n input gates in the family, therefore the computed mapping is in general only partially defined). In the literature on circuit complexity one can find more restrictive notions of uniformity, see e.g. [26], but logspace uniformity suffices for our purposes. In fact, polynomial time uniformity suffices for proving our lower bounds w.r.t. polynomial time reductions.

For $m \geq 1$ and $0 \leq M \leq 2^m - 1$ let $\mathrm{BIN}_m(M) = \mathrm{bit}_m(M) \cdots \mathrm{bit}_1(M) \in \{0,1\}^m$ denote the m-bit binary representation of M. Let p_i denote the ith prime number. It is well-known that the ith prime requires $O(\log(i))$ bits in its binary representation. For a number $0 \leq M < \prod_{i=1}^m p_i$ we define the *Chinese remainder representation* $\mathrm{CRR}_m(M)$ as the Boolean tuple $\mathrm{CRR}_m(M) = (x_{i,r})_{i\in[m], 0\leq r<p_i}$ with $x_{i,r} = 1$ if $M \bmod p_i = r$ and $x_{i,r} = 0$ else. By the following theorem, one can transform a Chinese remainder representation very efficiently into binary representation.

Theorem 6.1 ([9])**.** *There is a logspace-uniform* NC^1*-circuit family* $(B_m((x_{i,r})_{i\in[m],0\leq r<p_i}))_{m\geq 1}$ *such that for every* $m \geq 1$*,* B_m *has* m *output gates and for every* $0 \leq M < \prod_{i=1}^m p_i$ *we have that* $B_m(\mathrm{CRR}_m(M)) = \mathrm{BIN}_m(M \bmod 2^m)$*.*

By [17], we could replace logspace-uniform NC^1-circuits in Thm. 6.1 even by $\mathsf{DLOGTIME}$-uniform TC^0-circuits. The existence of a P-uniform NC^1-circuit family for converting from Chinese remainder representation to binary representation was already shown in [3]. Usually the Chinese remainder representation of M is the tuple $(r_i)_{i\in[m]}$, where $r_i = M \bmod p_i$. Since the primes p_i will be always given in unary notation, there is no essential difference between this representation and our Chinese remainder representation. The latter is more suitable for our purpose.

The following definition of NC^1-serializability is a variant of the more classical notion of serializability [8, 16], which fits our purpose better. A language L is NC^1*-serializable* if there exists an NFA A over the alphabet $\{0,1\}$, a polynomial $p(n)$, and a logspace-uniform NC^1-circuit family $(C_n)_{n\geq 0}$, where C_n has exactly $n+p(n)$ many inputs and one output, such that for every $x \in \{0,1\}^n$ we have $x \in L$ if and only if $C_n(x, 0^{p(n)}) \cdots C_n(x, 1^{p(n)}) \in L(A)$, where "$\cdots$" refers to the lexicographic order on $\{0,1\}^{p(n)}$. With this definition, it can be shown that all languages in PSPACE are NC^1-serializable. A proof can be found in the appendix of [14]; it is just a slight adaptation of the proofs from [8, 16].

7. Data complexity for CTL is hard for PSPACE

In this section, we prove that also the data complexity of CTL over OCNs is hard for PSPACE and therefore PSPACE-complete by the known upper bounds for the modal μ-calculus [23]. Let us fix the set of propositions $\mathcal{P} = \{\alpha, \beta, \gamma\}$ for this section. In the following, w.l.o.g. we allow in δ_0 (resp. in $\delta_{>0}$) transitions of the kind (q, k, q'), where $k \in \mathbb{N}$ (resp. $k \in \mathbb{Z}$) is given in unary representation with the expected intuitive meaning.

Proposition 7.1. *For the fixed* EF *formula* $\varphi = (\alpha \to \exists \mathsf{X}(\beta \wedge \mathsf{EF}(\neg \exists \mathsf{X} \gamma)))$ *the following problem can be solved with a logspace transducer:*
INPUT: A list $p_1, \ldots, p_m$ *of the first* m *consecutive (unary encoded) prime numbers and a Boolean formula* $F = F((x_{i,r})_{i \in [m], 0 \le r < p_i})$
OUTPUT: An OCN $\mathbb{O}(F)$ *with distinguished control locations* in *and* out*, such that for every number* $0 \le M < \prod_{i=1}^{m} p_i$ *we have that* $F(\mathrm{CRR}_m(M)) = 1$ *if and only if there exists a* $[\![\varphi]\!]_{T(\mathbb{O}(F))}$*-path from* (in, M) *to* (out, M) *in the transition system* $T(\mathbb{O}(F))$.

Proof. W.l.o.g., negations occur in F only in front of variables. Then additionally, a negated variable $\neg x_{i,r}$ can be replaced by the disjunction $\bigvee\{x_{i,k} \mid 0 \le k < p_i, r \ne k\}$. This can be done in logspace, since the primes p_i are given in unary. Thus, we can assume that F does not contain negations.

The idea is to traverse the Boolean formula F with the OCN $\mathbb{O}(F)$ in a depth first manner. Each time a variable $x_{i,r}$ is seen, the OCN may also enter another branch, where it is checked, whether the current counter value is congruent r modulo p_i. Let $\mathbb{O}(F) = (Q, \{Q_\alpha, Q_\beta, Q_\gamma\}, \delta_0, \delta_{>0})$, where $Q = \{\mathsf{in}(G), \mathsf{out}(G) \mid G \text{ is a subformula of } F\} \cup \{\mathrm{div}(p_1), \ldots, \mathrm{div}(p_m), \bot\}$, $Q_\alpha = \{\mathsf{in}(x_{i,r}) \mid i \in [m], 0 \le r < p_i\}$, $Q_\beta = \{\mathrm{div}(p_1), \ldots, \mathrm{div}(p_m)\}$, and $Q_\gamma = \{\bot\}$. We set $\mathsf{in} = \mathsf{in}(F)$ and $\mathsf{out} = \mathsf{out}(F)$. Let us now define the transition sets δ_0 and $\delta_{>0}$. For every subformula $G_1 \wedge G_2$ or $G_1 \vee G_2$ of F we add the following transitions to δ_0 and $\delta_{>0}$:

$$\mathsf{in}(G_1 \wedge G_2) \xrightarrow{0} \mathsf{in}(G_1),\ \mathsf{out}(G_1) \xrightarrow{0} \mathsf{in}(G_2),\ \mathsf{out}(G_2) \xrightarrow{0} \mathsf{out}(G_1 \wedge G_2)$$

$$\mathsf{in}(G_1 \vee G_2) \xrightarrow{0} \mathsf{in}(G_i),\ \mathsf{out}(G_i) \xrightarrow{0} \mathsf{out}(G_1 \vee G_2) \text{ for all } i \in \{1, 2\}$$

For every variable $x_{i,r}$ we add to δ_0 and $\delta_{>0}$ the transition $\mathsf{in}(x_{i,r}) \xrightarrow{0} \mathsf{out}(x_{i,r})$. Moreover, we add to $\delta_{>0}$ the transitions $\mathsf{in}(x_{i,r}) \xrightarrow{-r} \mathrm{div}(p_i)$. The transition $\mathsf{in}(x_{i,0}) \xrightarrow{0} \mathrm{div}(p_i)$ is also added to δ_0. For the control locations $\mathrm{div}(p_i)$ we add to $\delta_{>0}$ the transitions $\mathrm{div}(p_i) \xrightarrow{-p_i} \mathrm{div}(p_i)$ and $\mathrm{div}(p_i) \xrightarrow{-1} \bot$. This concludes the description of the OCN $\mathbb{O}(F)$. Correctness of the construction can be easily checked by induction on the structure of the formula F. ∎

We are now ready to prove PSPACE-hardness of the data complexity.

Theorem 7.2. *There exists a fixed* CTL *formula of the form* $\exists \varphi_1 \mathsf{U} \varphi_2$*, where* φ_1 *and* φ_2 *are* EF *formulas, for which it is* PSPACE*-complete to decide* $(T(\mathbb{O}), (q, 0)) \models \exists \varphi_1 \mathsf{U} \varphi_2$ *for a given OCN* $\mathbb{O}$ *and a control location* q *of* $\mathbb{O}$.

Proof. Let us take an arbitrary language L in PSPACE. Recall from Sec. 6 that PSPACE is NC^1-serializable. Thus, there exists an NFA $A = (S, \{0, 1\}, \delta, s_0, S_f)$ over the alphabet $\{0, 1\}$, a polynomial $p(n)$, and a logspace-uniform NC^1-circuit family $(C_n)_{n \ge 0}$, where C_n has $n + p(n)$ many inputs and one output, such that for every $x \in \{0, 1\}^n$ we have:

$$x \in L \iff C_n(x, 0^{p(n)}) \cdots C_n(x, 1^{p(n)}) \in L(A), \tag{7.1}$$

where "$\cdots$" refers to the lexicographic order on $\{0, 1\}^{p(n)}$. Fix an input $x \in \{0, 1\}^n$. Our reduction can be split into the following five steps:

Step 1. Construct in logspace the circuit C_n. Fix the the first n inputs of C_n to the bits in x, and denote the resulting circuit by C; it has only $m = p(n)$ many inputs. Then, (7.1) can be written as

$$x \in L \iff \prod_{M=0}^{2^m-1} C(\mathrm{BIN}_m(M)) \in L(A). \tag{7.2}$$

Step 2. Compute the first m consecutive primes $p_1, \ldots, p_m$. This is possible in logspace, see e.g. [9]. Every p_i is bounded polynomially in n. Hence, every p_i can be written down in unary notation. Note that $\prod_{i=1}^m p_i > 2^m$ (if $m > 1$).

Step 3. Compute in logspace the circuit $B = B_m((x_{i,r})_{i\in[m],0\le r<p_i})$ from Thm. 6.1. Thus, B is a Boolean circuit of fan-in 2 and depth $O(\log(m)) = O(\log(n))$ with m output gates and $B(\mathrm{CRR}_m(M)) = \mathrm{BIN}_m(M \bmod 2^m)$ for every $0 \le M < \prod_{i=1}^m p_i$.

Step 4. Now we compose the circuits B and C: For every $i \in [m]$, connect the ith input of the circuit $C(x_1, \ldots, x_m)$ with the ith output of the circuit B. The result is a circuit with fan-in 2 and depth $O(\log(n))$. In logspace, we can unfold this circuit into a Boolean formula $F = F((x_{i,r})_{i\in[m],0\le r<p_i})$. The resulting formula (or tree) has the same depth as the circuit, i.e., depth $O(\log(n))$ and every tree node has at most 2 children. Hence, F has polynomial size. For every $0 \le M < 2^m$ we have $F(\mathrm{CRR}_m(M)) = C(\mathrm{BIN}_m(M))$ and equivalence (7.2) can be written as

$$x \in L \iff \prod_{M=0}^{2^m-1} F(\mathrm{CRR}_m(M)) \in L(A). \tag{7.3}$$

Step 5. We now apply our construction from Prop. 7.1 to the formula F. More precisely, let G be the Boolean formula $\bigwedge_{i\in[m]} x_{i,r_i}$ where $r_i = 2^m \bmod p_i$ for $i \in [m]$ (these remainders can be computed in logspace). For every 1-labeled transition $\tau \in \delta$ of the NFA A let $\mathbb{O}(\tau)$ be a copy of the OCN $\mathbb{O}(F \wedge \neg G)$. For every 0-labeled transition $\tau \in \delta$ let $\mathbb{O}(\tau)$ be a copy of the OCN $\mathbb{O}(\neg F \wedge \neg G)$. In both cases we write $\mathbb{O}(\tau)$ as $(Q(\tau), \{Q_\alpha(\tau), Q_\beta(\tau), Q_\gamma(\tau)\}, \delta_0(\tau), \delta_{>0}(\tau))$. Denote with $\mathsf{in}(\tau)$ (resp. $\mathsf{out}(\tau)$) the control location of this copy that corresponds to in (resp. out) in $\mathbb{O}(F)$. Hence, for every b-labeled transition $\tau \in \delta$ ($b \in \{0,1\}$) and every $0 \le M < \prod_{i=1}^m p_i$ there exists a $[\![\varphi]\!]_{T(\mathbb{O}(\tau))}$-path ($\varphi$ is from Prop. 7.1) from $(\mathsf{in}(\tau), M)$ to $(\mathsf{out}(\tau), M)$ if and only if $F(\mathrm{CRR}_m(M)) = b$ and $M \ne 2^m$.

We now define an OCN $\mathbb{O} = (Q, \{Q_\alpha, Q_\beta, Q_\gamma\}, \delta_0, \delta_{>0})$ as follows: We take the disjoint union of all the OCNs $\mathbb{O}(\tau)$ for $\tau \in \delta$. Moreover, every state $s \in S$ of the NFA A becomes a control location of $\mathbb{O}$, i.e. $Q = S \cup \bigcup_{\tau\in\delta} Q(\tau)$ and $Q_p = \bigcup_{\tau\in\delta} Q_p(\tau)$ for each $p \in \{\alpha, \beta, \gamma\}$. We add to δ_0 and $\delta_{>0}$ for every $\tau = (s, b, t) \in \delta$ the transitions $s \xrightarrow{0} \mathsf{in}(\tau)$ and $\mathsf{out}(\tau) \xrightarrow{1} t$. Then, by Prop. 7.1 and (7.3) we have $x \in L$ if and only if there exists a $[\![\varphi]\!]_{T(\mathbb{O})}$-path in $T(\mathbb{O})$ from $(s_0, 0)$ to $(s, 2^m)$ for some $s \in S_f$. Also note that there is no $[\![\varphi]\!]_{T(\mathbb{O})}$-path in $T(\mathbb{O})$ from $(s_0, 0)$ to some configuration (s, M) with $s \in S$ and $M > 2^m$. It remains to add to $\mathbb{O}$ some structure that enables $\mathbb{O}$ to check that the counter has reached the value 2^m. For this, use again Prop. 7.1 to construct the OCN $\mathbb{O}(G)$ (G is from above) and add it disjointly to $\mathbb{O}$. Moreover, add to $\delta_{>0}$ and δ_0 the transitions $s \xrightarrow{0} \mathsf{in}$ for all $s \in S_f$, where in is the in control location of $\mathbb{O}(G)$. Finally, introduce a new proposition ρ and set $Q_\rho = \{\mathsf{out}\}$, where out is the out control location of $\mathbb{O}(G)$. By putting $q = s_0$ we obtain: $x \in L$ if and only if $(T(\mathbb{O}), (q, 0)) \models \exists(\varphi \mathsf{U} \rho)$, where φ is from Prop. 7.1. This concludes the proof of the theorem. ∎

By slightly modifying the proof of Thm. 7.2, one can also prove that the fixed CTL formula can chosen to be of the form $\exists\mathsf{G}\psi$, where ψ is an EF formula.

8. Two further applications: EF and one-counter Markov decision processes

In this section, we present two further applications of Thm. 6.1 to OCPs. First, we state that the combined complexity for EF over OCNs is hard for P^{NP}. For formulas represented succinctly by directed acyclic graphs this was already shown in [13]. The point here is that we use the standard tree representation for formulas.

Theorem 8.1. *It is* P^{NP}*-hard (and hence* P^{NP}*-complete by* [13]*) to check* $(T(\mathbb{O}),(q_0,0)) \models \varphi$ *for given OCN* $\mathbb{O}$*, state* q_0 *of* $\mathbb{O}$*, and* EF *formula* φ*.*

The proof of Thm. 8.1 is very similar to the proof of Thm. 7.2, but does not use the concept of serializability. We prove hardness by a reduction from the question whether the lexicographically maximal satisfying assignment of a Boolean formula is even when interpreted as a natural number. This problem is P^{NP}-hard by [27]. At the moment we cannot prove that the data complexity of EF over OCPs is hard for P^{NP} (hardness for $\mathsf{P}^{\mathsf{NP}[\log]}$ was shown in [13]). Analyzing the proof of Thm. 8.1 in [14] shows that the main obstacle is the fact that converting from Chinese remainder representation into binary representation is not possible by uniform AC^0 circuits (polynomial size circuits of constant depth and unbounded fan-in); this is provably the case.

In the rest of the paper, we sketch a second application of our lower bound technique based on Thm. 6.1, see [14] for more details. This application concerns one-counter Markov decision processes. *Markov decision processes* (MDPs) extend classical Markov chains by allowing so called *nondeterministic vertices*. In these vertices, no probability distribution on the outgoing transitions is specified. The other vertices are called *probabilistic vertices*; in these vertices a probability distribution on the outgoing transitions is given. The idea is that in an MDP a player Eve plays against nature (represented by the probabilistic vertices). In each nondeterministic vertex v, Eve chooses a probability distribution on the outgoing transitions of v; this choice may depend on the past of the play (which is a path in the underlying graph ending in v) and is formally represented by a strategy for Eve. An MDP together with a strategy for Eve defines a Markov chain, whose state space is the unfolding of the graph underlying the MDP. Here, we consider infinite MDPs, which are finitely represented by OCPs; this formalism was introduced in [6] under the name *one-counter Markov decision process* (OC-MDP). With a given OC-MDP $\mathcal{A}$ and a set R of control locations of the OCP underlying $\mathcal{A}$ (a so called *reachability constraint*), two sets were associated in [6]: $\mathrm{ValOne}(R)$ is the set of all vertices s of the MDP defined by $\mathcal{A}$ such that for every $\epsilon > 0$ there exists a strategy σ for Eve under which the probability of finally reaching from s a control location in R and at the same time having counter value 0 is at least $1-\varepsilon$. $\mathrm{OptValOne}(R)$ is the set of all vertices s of the MDP defined by $\mathcal{A}$ for which there exists a specific strategy for Eve under which this probability is 1. It was shown in [6] that for a given OC-MDP $\mathcal{A}$, a set of control locations R, and a vertex s of the MDP defined by $\mathcal{A}$, the question if $s \in \mathrm{OptValOne}(R)$ is PSPACE-hard and in $\mathsf{EXPTIME}$. The same question for $\mathrm{ValOne}(R)$ instead of $\mathrm{OptValOne}(R)$ was shown to be hard for each level of the Boolean hierarchy BH, which is a hierarchy of complexity classes between NP and $\mathsf{P}^{\mathsf{NP}[\log]}$. By applying our lower bound techniques (from Thm. 7.2) we can prove the following.

Theorem 8.2. *Membership in ValOne*(R) *is* PSPACE*-hard.*

As a byproduct of our proof, we also reprove PSPACE-hardness for $\mathrm{OptValOne}(R)$. It is open, whether $\mathrm{ValOne}(R)$ is decidable; the corresponding problem for MDPs defined by pushdown processes is undecidable [11].

References

[1] S. Arora and B. Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.

[2] C. Baier and J. P. Katoen. *Principles of Model Checking*. MIT Press, 2009.

[3] P. W. Beame, S. A. Cook, and H. J. Hoover. Log depth circuits for division and related problems. *SIAM J. Comput.*, 15(4):994–1003, 1986.

[4] A. Bouajjani, J. Esparza, and O. Maler. Reachability analysis of pushdown automata: Application to model-checking. In *Proc. CONCUR'97*, LNCS 1243, 135–150. Springer, 1997.

[5] L. Bozzelli. Complexity results on branching-time pushdown model checking. *T. Comput. Sci.*, 379:286–297, 2007.

[6] T. Brazdil, V. Brozek, K. Etessami, A. Kucera, and D. Wojtczak. One-counter markov decision processes. *Proc. SODA 2010*, 863–874. SIAM, 2010.

[7] T. Cachat. Uniform solution of parity games on prefix-recognizable graphs. *ENTCS*, 68(6):71–84, 2002.

[8] J.-Y. Cai and M. Furst. PSPACE survives constant-width bottlenecks. *Internat. J. Found. Comput. Sci.*, 2(1):67–76, 1991.

[9] A. Chiu, G. Davida, and B. Litow. Division in logspace-uniform NC^1. *RAIRO Inform. Théor. Appl.*, 35(3):259–275, 2001.

[10] J. Esparza, D. Hansel, P. Rossmanith, and S. Schwoon. Efficient algorithms for model checking pushdown systems. In *Proc. CAV 2000*, LNCS 1855, 232–247. Springer, 2000.

[11] K. Etessami and M. Yannakakis. Recursive markov decision processes and recursive stochastic games. In *Proc. ICALP 2005*, LNCS 3580, 891–903. Springer, 2005.

[12] S. Göller and M. Lohrey. Infinite state model-checking of propositional dynamic logics. In *Proc. CSL 2006*, LNCS 4207, 349–364. Springer, 2006.

[13] S. Göller, R. Mayr, and A. W. To. On the computational complexity of verifying one-counter processes. In *Proc. LICS 2009*, 235–244. IEEE Computer Society Press, 2009.

[14] S. Göller, M. Lohrey. Branching-time model checking of one-counter processes. `http://arxiv.org/abs/0909.1102`.

[15] C. Haase, S. Kreutzer, J. Ouaknine, and J. Worrell. Reachability in succinct and parametric one-counter automata. In *Proc. CONCUR'09*, LNCS 5710, 369–383. Springer, 2009.

[16] U. Hertrampf, C. Lautemann, T. Schwentick, H. Vollmer, and K. W. Wagner. On the power of polynomial time bit-reductions. In *Proc. 8th Annual Structure in Complexity Theory Conference*, 200–207. IEEE Computer Society Press, 1993.

[17] W. Hesse, E. Allender, and D. A. M. Barrington. Uniform constant-depth threshold circuits for division and iterated multiplication. *J. Comput. System Sci.*, 65:695–716, 2002.

[18] M. Holzer. On emptiness and counting for alternating finite automata. In *Proc. DLT 1995*, 88–97. Wo. Scient., 1996.

[19] P. Jančar and Z. Sawa. A note on emptiness for alternating finite automata with a one-letter alphabet. *I. P. L.*, 104(5):164–167, 2007.

[20] M. Nair. On Chebyshev-type inequalities for primes. *Amer. Math. Monthly*, 89(2):126–129, 1982.

[21] N. Piterman and M. Y. Vardi. Global model-checking of infinite-state systems. In *Proc. CAV 2004*, LNCS 3114, 387–400. Springer, 2004.

[22] O. Serre. Note on winning positions on pushdown games with ω-regular conditions. *I. P. L.*, 85(6):285–291, 2003.

[23] O. Serre. Parity games played on transition graphs of one-counter processes. In *Proc. FOSSACS 2006*, LNCS 3921, 337–351. Springer, 2006.

[24] D. Thérien and T. Wilke. Temporal logic and semidirect products: An effective characterization of the until hierarchy. In *Proc. FOCS '96*, 256–263. IEEE Computer Society Press, 1996.

[25] A. W. To. Model checking FO(R) over one-counter processes and beyond. In *Proc. CSL 2009*, LNCS 5771, 485–499. Springer, 2009.

[26] H. Vollmer. *Introduction to Circuit Complexity*. Springer, 1999.

[27] K. W. Wagner. More complicated questions about maxima and minima, and some closures of NP. *Theoret. Comput. Sci.*, 51:53–80, 1987.

[28] I. Walukiewicz. Model checking CTL properties of pushdown systems. In *Proc. FSTTCS 2000*, LNCS 1974, 127–138. Springer, 2000.

[29] I. Walukiewicz. Pushdown processes: games and model-checking. *Inform. and Comput.*, 164(2):234–263, 2001.

This work is licensed under the Creative Commons Attribution-NoDerivs License. To view a copy of this license, visit `http://creativecommons.org/licenses/by-nd/3.0/`.