# Bisimilarity of Probabilistic Pushdown Automata

Vojtěch Forejt[1], Petr Jančar[2], Stefan Kiefer[1], and James Worrell[1]

1   Department of Computer Science, University of Oxford, UK
2   Department of Computer Science, FEI, Techn. Univ. Ostrava, Czech Republic

### Abstract

We study the bisimilarity problem for probabilistic pushdown automata (pPDA) and subclasses thereof. Our definition of pPDA allows both probabilistic and non-deterministic branching, generalising the classical notion of pushdown automata (without $\varepsilon$-transitions). Our first contribution is a general construction that reduces checking bisimilarity of probabilistic transition systems to checking bisimilarity of non-deterministic transition systems. This construction directly yields decidability of bisimilarity for pPDA, as well as an elementary upper bound for the bisimilarity problem on the subclass of probabilistic basic process algebras, i.e., single-state pPDA. We further show that, with careful analysis, the general reduction can be used to prove an EXPTIME upper bound for bisimilarity of probabilistic visibly pushdown automata. Here we also provide a matching lower bound, establishing EXPTIME-completeness. Finally we prove that deciding bisimilarity of probabilistic one-counter automata, another subclass of pPDA, is PSPACE-complete. Here we use a more specialised argument to obtain optimal complexity bounds.

## 1   Introduction

Equivalence checking is the problem of determining whether two systems are semantically identical. This is an important question in automated verification and, more generally, represents a line of research that can be traced back to the inception of theoretical computer science. A great deal of work in this area has been devoted to the complexity of *bisimilarity* for various classes of infinite-state systems related to grammars, such as one-counter automata, basic process algebras, and pushdown automata, see [4] for an overview. We mention in particular the landmark result showing the decidability of bisimilarity for pushdown automata [14].

In this paper we are concerned with probabilistic pushdown automata (pPDA), that is, pushdown automata with both non-deterministic and probabilistic branching. In particular, our pPDA generalise classical pushdown automata without $\varepsilon$-transitions. We refer to automata with only probabilistic branching as *fully probabilistic*.

We consider the complexity of checking bisimilarity for probabilistic pushdown automata and various subclasses thereof. The subclasses we consider are probabilistic versions of models that have been extensively studied in previous works [4, 15]. In particular, we consider probabilistic one-counter automata (pOCA), which are probabilistic pushdown automata with singleton stack alphabet; probabilistic Basic Process Algebras (pBPA), which are single-state probabilistic pushdown automata; probabilistic visibly pushdown automata (pvPDA), which are automata in which the stack action, whether to push or pop, for each transition is determined by the input letter. Probabilistic one-counter automata have been studied in the classical theory of stochastic processes as *quasi-birth-death processes* [6]. Probabilistic BPA seems to have been introduced in [3].

While the complexity of bisimilarity for finite-state probabilistic automata is well understood [1, 5], there are relatively few works on equivalence of infinite-state probabilistic systems. Bisimilarity of probabilistic BPA was shown decidable in [3], but without any complexity bound. In [7] probabilistic simulation between probabilistic pushdown automata and finite state systems was studied.

## 1.1 Contribution

The starting point of the paper is a construction that can be used to reduce the bisimilarity problem for many classes of probabilistic systems to the bisimilarity problem for their non-probabilistic counterparts. The reduction relies on the observation that in the bisimilarity problem, the numbers that occur as probabilities in a probabilistic system can be "encoded" as actions in the non-probabilistic system. This comes at the price of an exponential blow-up in the branching size, but still allows us to establish several new results. It is perhaps surprising that there is a relatively simple reduction of probabilistic bisimilarity to ordinary bisimilarity. Hitherto it has been typical to establish decidability in the probabilistic case using bespoke proofs, see, e.g., [3, 7]. Instead, using our reduction, we can leverage the rich theory that has been developed in the non-probabilistic case.

The main results of the paper are as follows:

- Using the above-mentioned reduction together with the result of [14], we show that bisimilarity for probabilistic pushdown automata is decidable.
- For the subclass of probabilistic BPA, i.e., automata with a single control state, the same reduction yields a 3EXPTIME upper bound for checking bisimilarity via a doubly exponential procedure for bisimilarity on BPA [4] (see also [10]). This improves the result of [3], where only a decidability result was given without any complexity bound. An EXPTIME lower bound for this problem follows from the recent work of [11] for non-probabilistic systems.
- For probabilistic visibly pushdown automata, the above reduction immediately yields a 2EXPTIME upper bound. However we show that with more careful analysis we can extract an EXPTIME upper bound. In this case we also show EXPTIME-hardness, thus obtaining matching lower and upper bounds.
- For fully probabilistic one-counter automata we obtain matching lower and upper PSPACE bounds for the bisimilarity problem. In both cases the bounds are obtained by adapting constructions from the non-deterministic case [15, 2] rather than by using the generic reduction described above.

## 2 Preliminaries

Given a countable set $A$, a *probability distribution* on $A$ is a function $d : A \to [0,1] \cap \mathbb{Q}$ (the rationals) such that $\sum_{a \in A} d(a) = 1$. A probability distribution is *Dirac* if it assigns 1 to one element and 0 to all the others. The *support* of a probability distribution $d$ is the set $support(d) := \{a \in A : d(a) > 0\}$. The set of all probability distributions on $A$ is denoted by $\mathcal{D}(A)$.

## 2.1 Probabilistic Transition Systems.

A *probabilistic labelled transition system (pLTS)* is a tuple $\mathcal{S} = (S, \Sigma, \to)$, where $S$ is a finite or countable set of *states*, $\Sigma$ is a finite input *alphabet*, and $\to \subseteq S \times \Sigma \times \mathcal{D}(S)$ is a *transition relation*. We write $s \xrightarrow{a} d$ to say that $(s, a, d) \in \to$. We also write $s \to s'$ to say that

there exists $s \xrightarrow{a} d$ with $s' \in support(d)$. We assume that $\mathcal{S}$ is finitely branching, i.e., each state $s$ has finitely many transitions $s \xrightarrow{a} d$. In general a pLTS combines probabilistic and non-deterministic branching. A pLTS is said to be *fully probabilistic* if for each state $s \in S$ and action $a \in \Sigma$ we have $s \xrightarrow{a} d$ for at most one distribution $d$. Given a fully probabilistic pLTS, we write $s \xrightarrow{a,x} s'$ to say that there is $s \xrightarrow{a} d$ such that $d(s') = x$.

Let $\mathcal{S} = (S, \Sigma, \rightarrow)$ be a pLTS and $R$ an equivalence relation on $S$. We say that two distributions $d, d' \in \mathcal{D}(S)$ are *R-equivalent* if for all $R$-equivalence classes $E$, $\sum_{s \in E} d(s) = \sum_{s \in E} d'(s)$. We furthermore say that $R$ is a *bisimulation relation* if $s \, R \, t$ implies that for each action $a \in \Sigma$ and each transition $s \xrightarrow{a} d$ there is a transition $t \xrightarrow{a} d'$ such that $d$ and $d'$ are $R$-equivalent. The union of all bisimulation relations of $\mathcal{S}$ is itself a bisimulation relation. This relation is called *bisimilarity* and is denoted $\sim$ [13].

We also have the following inductive characterisation of bisimilarity. Define a decreasing sequence of equivalence relations $\sim_0 \supseteq \sim_1 \supseteq \sim_2 \supseteq \cdots$ by putting $s \sim_0 t$ for all $s, t$, and $s \sim_{n+1} t$ if and only if for all $a \in \Sigma$ and $s \xrightarrow{a} d$ there is $t \xrightarrow{a} d'$ such that $\sum_{s \in E} d(s) = \sum_{s \in E} d'(s)$ for all $\sim_n$-equivalence classes $E$. It is then straightforward that the sequence $\sim_n$ converges to $\sim$, i.e., $\bigcap_{n \in \mathbb{N}} \sim_n = \sim$.

## 2.2 Probabilistic Pushdown Automata.

A *probabilistic pushdown automaton* (pPDA) is a tuple $\Delta = (Q, \Gamma, \Sigma, \hookrightarrow)$ where $Q$ is a finite set of *states*, $\Gamma$ is a finite *stack alphabet*, $\Sigma$ is a finite *input alphabet*, and $\hookrightarrow \subseteq Q \times \Gamma \times \Sigma \times \mathcal{D}(Q \times \Gamma^{\leq 2})$ (with $\Gamma^{\leq 2} := \{\varepsilon\} \cup \Gamma \cup (\Gamma \times \Gamma)$) (where $\varepsilon$ denotes the empty string).

When speaking of the *size* of $\Delta$, we assume that the probabilities in the transition relation are given as quotients of integers written in binary. A tuple $(q, X) \in Q \times \Gamma$ is called a *head*. A pPDA is *fully probabilistic* if for each head $(q, X)$ and action $a \in \Sigma$ there is at most one distribution $d$ with $(q, X, a, d) \in \hookrightarrow$. A configuration of a pPDA is an element $(q, \beta) \in Q \times \Gamma^*$, and we sometimes write just $q\beta$ instead of $(q, \beta)$. We write $qX \xrightarrow{a} d$ to denote $(q, X, a, d) \in \hookrightarrow$, that is, in a control state $q$ with $X$ at the top of the stack the pPDA makes an $a$-transition to the distribution $d$. In a fully probabilistic pPDA we also write $qX \xrightarrow{a,x} r\beta$ if $qX \xrightarrow{a} d$ and $d(r\beta) = x$.

A *probabilistic basic process algebra (pBPA)* $\Delta$ is a pPDA with only one control state. In this case we sometimes omit the control state from the representation of a configuration. A *probabilistic one-counter automaton (pOCA)* is a pPDA with a stack alphabet containing only two symbols $X$ and $Z$, where the transition function is restricted so that $Z$ always and only occurs at the bottom of the stack. A *probabilistic visibly pushdown automaton (pvPDA)* is a pPDA with a partition of the actions $\Sigma = \Sigma_r \cup \Sigma_{int} \cup \Sigma_c$ such that for all $pX \xrightarrow{a} d$ we have: if $a \in \Sigma_r$ then $support(d) \subseteq Q \times \{\varepsilon\}$; if $a \in \Sigma_{int}$ then $support(d) \subseteq Q \times \Gamma$; if $a \in \Sigma_c$ then $support(d) \subseteq Q \times (\Gamma \times \Gamma)$.

A pPDA $\Delta = (Q, \Gamma, \Sigma, \hookrightarrow)$ generates a pLTS $\mathcal{S}(\Delta) = (Q \times \Gamma^*, \Sigma, \rightarrow)$ as follows. For each $\beta \in \Gamma^*$ a rule $qX \xrightarrow{a} d$ of $\Delta$ induces a transition $qX\beta \xrightarrow{a} d'$ in $\mathcal{S}(\Delta)$, where $d' \in \mathcal{D}(Q \times \Gamma^*)$ is defined by $d'(p\alpha\beta) = d(p\alpha)$ for all $p \in Q$ and $\alpha \in \Gamma^*$. Note that all configurations with the empty stack define terminating states of $\mathcal{S}(\Delta)$.

The *bisimilarity problem* asks whether two configurations $q_1\alpha_1$ and $q_2\alpha_2$ of a given pPDA $\Delta$ are bisimilar when regarded as states of the induced pLTS $\mathcal{S}(\Delta)$.

▶ **Example 1.** Consider the fully probabilistic pPDA $\Delta = (\{p, q, r\}, \{X, X', Y, Z\}, \{a\}, \hookrightarrow)$
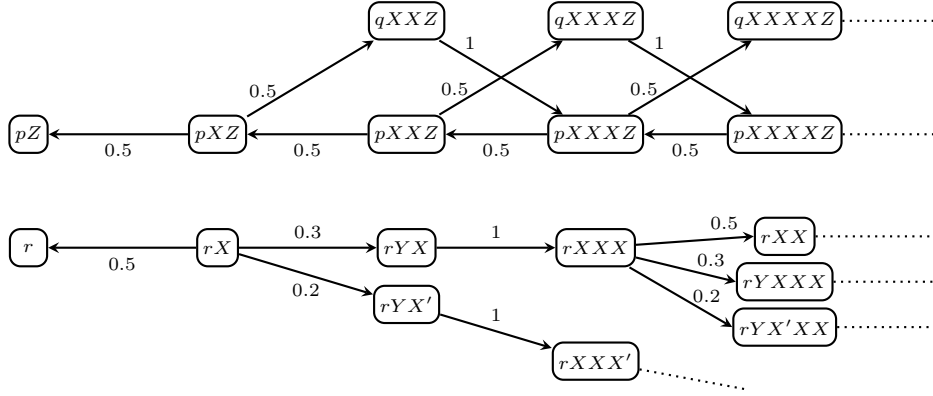
**Figure 1** A fragment of $\mathcal{S}(\Delta)$ from Example 1.

with the following rules (omitting the unique action $a$):

$$pX \stackrel{0.5}{\hookrightarrow} qXX, \qquad\qquad\qquad pX \stackrel{0.5}{\hookrightarrow} p, \qquad qX \stackrel{1}{\hookrightarrow} pXX,$$

$$rX \stackrel{0.3}{\hookrightarrow} rYX, \qquad rX \stackrel{0.2}{\hookrightarrow} rYX', \qquad rX \stackrel{0.5}{\hookrightarrow} r, \qquad rY \stackrel{1}{\hookrightarrow} rXX,$$

$$rX' \stackrel{0.4}{\hookrightarrow} rYX, \qquad rX' \stackrel{0.1}{\hookrightarrow} rYX', \qquad rX' \stackrel{0.5}{\hookrightarrow} r.$$

The restriction of $\Delta$ to the control states $p, q$ and to the stack symbols $X, Z$ yields a pOCA. The restriction of $\Delta$ to the control state $r$ and the stack symbols $X, X', Y$ yields a pBPA. A fragment of the pLTS $\mathcal{S}(\Delta)$ is shown in Figure 1. The configurations $pXZ$ and $rX$ are bisimilar, as there is a bisimulation relation with equivalence classes $\{pX^k Z\} \cup \{rw \mid w \in \{X, X'\}^k\}$ for all $k \geq 0$ and $\{qX^{k+1}Z\} \cup \{rYw \mid w \in \{X, X'\}^k\}$ for all $k \geq 1$.

## 3 From Probabilistic to Nondeterministic Bisimilarity

A nondeterministic pushdown automaton (PDA) is a special case of a probabilistic pushdown automaton in which the transition function assigns only Dirac distributions. We give a novel reduction of the bisimilarity problem for pPDA to the bisimilarity problem for PDA. Because the latter is known to be decidable [14], we get decidability of the bisimilarity problem for pPDA.

As a first step we give the following characterisation of $R$-equivalence of two distributions (defined earlier).

▶ **Lemma 2.** *Let $R$ be an equivalence relation on a set $S$. Two distributions $d, d'$ on $S$ are $R$-equivalent if and only if for all $A \subseteq S$ we have $d(A) \leq d'(R(A))$, where $R(A)$ denotes the image of $A$ under $R$.*

**Proof.** For the *if* direction we reason as follows. For each equivalence class $E$ we have $d(E) \leq d'(E)$. But since $d$ and $d'$ have total mass 1 we must have $d(E) = d'(E)$ for all equivalence classes $E$.

Conversely if $d$ and $d'$ are $R$-equivalent. Then $d(A) \leq d(R(A)) = d'(R(A))$ for any set $A$, since $R(A)$ is a countable union of equivalence classes. ◀

We now give our reduction. Let $\Delta = (Q, \Gamma, \Sigma, \hookrightarrow)$ be a pPDA and $q_1\gamma_1$, $q_2\gamma_2$ two configurations of $\Delta$. We define a new PDA $\Delta' = (Q, \Gamma', \Sigma', \rightmapsto)$ that extends $\Delta$ with extra stack symbols, input letters and transition rules. In particular, a configuration of $\Delta$ can also

be regarded as a configuration of $\Delta'$. The definition of $\Delta'$ is such that two $\Delta$-configurations $q_1\gamma_1$ and $q_2\gamma_2$ are bisimilar in $\Delta$ if and only if the same two configurations are bisimilar in $\Delta'$.

Intuitively we eliminate probabilistic transitions by treating probabilities as part of the input alphabet. To this end, let $W \subseteq \mathbb{Q}$ be the set of rational numbers of the form $d(A)$ for some rule $pX \xrightarrow{a} d$ in $\Delta$ and $A \subseteq support(d)$. Think of $W$ as the set of relevant transition weights.

We define $\Delta'$ as follows. Note that when defining rules of $\Delta'$ we write just $q\gamma$ instead of the Dirac distribution assigning 1 to $q\gamma$.

- The stack alphabet $\Gamma'$ contains all symbols from $\Gamma$. In addition, for every rule $pX \xrightarrow{a} d$ in $\Delta$ it contains a new symbol $\langle d \rangle$ and for every $T \subseteq support(d)$ a symbol $\langle T \rangle$.
- The input alphabet $\Sigma'$ is equal to $\Sigma \cup W \cup \{\#\}$ where $\#$ is a distinguished action not in $\Sigma$ or $W$.
- The transition function $\circ\!\!\rightarrow$ is defined as follows. For every rule $qX \xrightarrow{a} d$, there is a rule $qX \xrightarrow{a} q\langle d \rangle$. We also have a rule $q\langle d \rangle \xrightarrow{w} q\langle T \rangle$ if $T \subseteq support(d)$ and $d(T) \geq w \in W$. Finally, we have a rule $q\langle T \rangle \xrightarrow{\#} p\alpha$ if $p\alpha \in T$.

The PDA $\Delta'$ can be constructed in time exponential in the size of $\Delta$, and in polynomial time if the branching degree of $\Delta$ is bounded (i.e. if we fix a number $N$ and consider only pP-DAs with branching degree at most $N$). See Appendix A.2 for the analysis. The correctness of the construction is captured by the following lemma and proved in Appendix A.1.

▶ **Lemma 3.** *For any configurations $q_1\gamma_1, q_2\gamma_2$ of $\Delta$ we have $q_1\gamma_1 \sim q_2\gamma_2$ in $\Delta$ if and only if $q_1\gamma_1 \sim q_2\gamma_2$ in $\Delta'$.*

Let us show intuitively why bisimilar configurations in $\Delta$ remain bisimilar considered as configurations of $\Delta'$. Every computation step of $\Delta$ is simulated in three steps by $\Delta'$. Let $q_1X_1\gamma_1$ and $q_2X_2\gamma_2$ be bisimilar configurations of $\Delta$. Then in $\Delta'$ a transition of $q_1X_1\gamma_1$ to $q_1\langle d_1 \rangle\gamma_1$ can be matched by a transition (under the same action) of $q_2X_2\gamma_2$ to $q_2\langle d_2 \rangle\gamma_2$ such that the distributions $d_1$ and $d_2$ are $\sim$-equivalent (and *vice versa*). In particular, by Lemma 2, for any set of configurations $T$ the set $T'$ obtained by saturating $T$ under bisimilarity is such that $d_1(T) \leq d_2(T')$. Let $\bar{T}$ and $\bar{T}'$ respectively contain the elements of $T$ and $T'$ from which the suffixes $\gamma_1$ and $\gamma_2$ are removed. Then, as a second step of simulation of $\Delta$ by $\Delta'$, a transition of $q_1\langle d_1 \rangle\gamma_1$ to a state $q_1\langle \bar{T} \rangle\gamma_1$ with label $w \in W$ can be matched by a transition of $\Delta'$ to $q_2\langle \bar{T}' \rangle\gamma_2$ with the same label (similarly any transition of $q_2\langle d_2 \rangle\gamma_2$ can be matched by a transition of $q_1\langle d_1 \rangle\gamma_1$). Finally, as $T$ and $T'$ contain elements from the same bisimilarity equivalence classes, in the third step a $\#$-transition from $q_1\langle \bar{T} \rangle\gamma_1$ to some $q_1'\alpha_1\gamma_1$ can be matched by a $\#$-transition of $q_2\langle \bar{T}' \rangle\gamma_2$ to $q_2'\alpha_2\gamma_2$ such that $q_1'\alpha_1\gamma_1$ and $q_2'\alpha_2\gamma_2$ are again bisimilar in $\Delta$ (and *vice versa*).

The three steps are illustrated in Figure 2, where the successors of the configurations $pXZ$ and $rX$ in the system $\mathcal{S}(\Delta')$ for the PDA $\Delta'$ constructed from the pPDA $\Delta$ from Example 1 are drawn.

Lemma 3 gives rise to the following theorem.

▶ **Theorem 4.** *For any pPDA $\Delta$ there is a PDA $\Delta'$ constructible in exponential time such that for any configurations $q_1\gamma_1, q_2\gamma_2$ of $\Delta$ we have $q_1\gamma_1 \sim q_2\gamma_2$ in $\Delta$ if and only if $q_1\gamma_1 \sim q_2\gamma_2$ in $\Delta'$. In addition, if $\Delta$ is a pBPA, then $\Delta'$ is a BPA.*

Using Theorem 4 and [14, 4], we get the following corollary.

▶ **Corollary 5.** *The bisimilarity problem for pPDA is decidable, and the bisimilarity problem for pBPA is decidable in triply exponential time.*
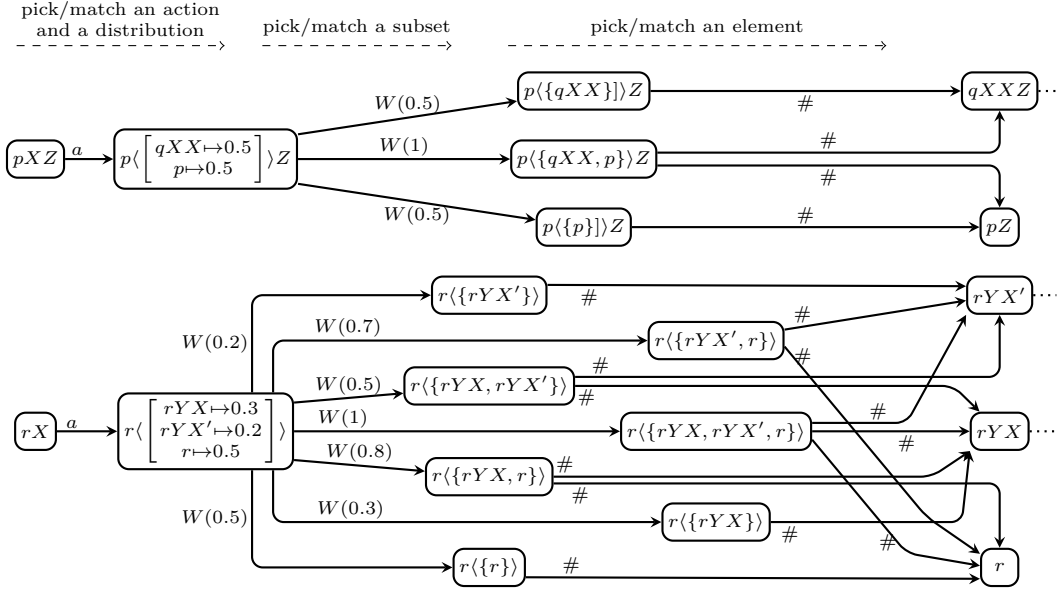
**Figure 2** An example of the construction for Lemma 3. Here, an arrow labelled $W(x)$ is an abbreviation for multiple transitions labelled by all multiples of 0.1 between 0.1 and $x$.

# 4 Upper Bounds

## 4.1 Bisimilarity of pOCA is in PSPACE

The bisimilarity problem for (non-probabilistic) one-counter automata is PSPACE-complete, as shown in [2]. It turns out that for pOCA we get PSPACE-completeness as well. The lower bound is shown in Section 5; here we show:

▶ **Theorem 6.** *The bisimilarity problem for pOCA is in PSPACE, even if we present the instance $\Delta = (Q, \{Z, X\}, \Sigma, \hookrightarrow)$, $pX^m Z, qX^n Z$ (for which we ask if $pX^m Z \sim qX^n Z$) by a shorthand using $m, n$ written in binary.*
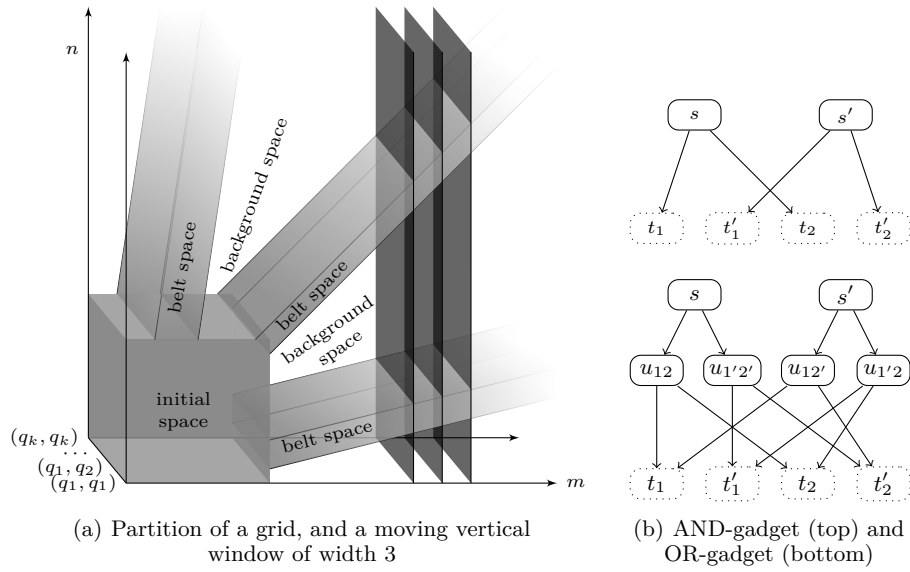
The reduction underlying Theorem 4 would only provide an exponential-space upper bound, so we give a pOCA-specific polynomial-space algorithm. In fact, we adapt the algorithm from [2]; the principles are the same but some ingredients have to be slightly modified. The following text is meant to give the idea in a self-contained manner, though at a more abstract level than in [2]. The main difference is in the notion of local consistency, discussed around Proposition 11.

Similarly as [2], we use a geometrical presentation of relations on the set of configurations (Fig. 3(a) reflects such a presentation). A relation can be identified with a 1/0 (or YES/NO) colouring of the "grid" $\mathbb{N} \times \mathbb{N} \times (Q \times Q)$:

▶ **Definition 7.** For a relation $R$ on $Q \times (\{X\}^* Z)$, by the *(characteristic) colouring $\chi_R$* we mean the function $\chi_R : \mathbb{N} \times \mathbb{N} \times (Q \times Q) \to \{1, 0\}$ where $\chi_R(m, n, (p, q)) = 1$ if and only if $(pX^m Z, qX^n Z) \in R$. Given (a colouring) $\chi : \mathbb{N} \times \mathbb{N} \times (Q \times Q) \to \{1, 0\}$, by $R_\chi$ we denote the relation $R_\chi = \{(pX^m Z, qX^n Z) \mid \chi(m, n, (p, q)) = 1\}$.

The algorithm uses the fact that $\chi_\sim$ is "regular", i.e. $\{(m, n, (p, q)) \mid pX^m Z \sim qX^n Z\}$ is a (special) semilinear set. More concretely, there are polynomials $pol_1, pol_2 : \mathbb{N} \to \mathbb{N}$ (independent of the pOCA $\Delta$) such that the following partition of the grid $\mathbb{N} \times \mathbb{N} \times (Q \times Q)$

(a) Partition of a grid, and a moving vertical window of width 3

(b) AND-gadget (top) and OR-gadget (bottom)

**Figure 3** Figures for Section 4.1 (left) and 5 (right)

(sketched in Fig. 3(a)) has an important property specified later. If $Q = \{q_1, q_2, \ldots, q_k\}$, hence $|Q| = k$, then the grid is partitioned into three parts: the *initial-space*, i.e. $\{(m, n, (p, q)) \mid m, n \leq pol_2(k)\}$, the *belt-space*, which is given by at most $k^4$ linear belts, with the slopes $\frac{c}{d}$ where $c, d \in \{1, 2, \ldots, k^2\}$ and with the (vertical) thickness bounded by $pol_1(k)$, and the rest, called the *background*. Moreover, $pol_2(k)$ is sufficiently large w.r.t. $pol_1(k)$, so that the belts are separated by the background outside the initial space.

The mentioned important property is that there is a period $\psi$, given by an exponential function of $k$, such that if two points $(m, n, (p, q))$ and $(m + i\psi, n + j\psi, (p, q))$ (for $i, j \in \mathbb{N}$) are both in the background, for both $m, n$ larger then a polynomial bound, then $\chi_\sim$ has the same value for both these points; in other words, $\chi_\sim$ colours the background periodically. Another important ingredient is the locality of the bisimulation conditions, resulting from the fact that the counter value can change by at most 1 per step.

To explain the "grid-partition", we start with considering the finite automaton $\mathcal{F}_\Delta$ underlying $\Delta$; $\mathcal{F}_\Delta$ behaves like $\Delta$ "pretending" that the counter is always positive.

▶ **Definition 8.** For a pOCA $\Delta = (Q, \{Z, X\}, \Sigma, \hookrightarrow)$, in the *underlying finite pLTS* $\mathcal{F}_\Delta = (Q, \Sigma, \rightarrow)$ we have a transition $p \xrightarrow{a} d'$ if and only if there is a transition $pX \xrightarrow{a} d$ such that $d'(q) = d(q, \varepsilon) + d(q, X) + d(q, XX)$ (for all $q \in Q$).

Using standard partition-refinement arguments, we observe that $\sim_{k-1} = \sim_k = \sim$ on $\mathcal{F}_\Delta$ when $k = |Q|$. For configurations of $\Delta$ we now define the distance $\mathsf{dist}$ to the set of configurations which are "INCompatible" with $\mathcal{F}_\Delta$.

▶ **Definition 9.** Assuming a pOCA $\Delta = (Q, \{Z, X\}, \Sigma, \hookrightarrow)$, where $|Q| = k$, we define $\mathsf{INC} \subseteq Q \times (\{X\}^* Z)$ and $\mathsf{dist} : Q \times (\{X\}^* Z) \rightarrow \mathbb{N} \cup \{\infty\}$ as follows:
- $\mathsf{INC} = \{pX^m Z \mid \forall q \in Q : pX^m Z \not\sim_k q\}$ (where $q$ is a state in $\mathcal{F}_\Delta$),
- $\mathsf{dist}(pX^m Z) = \min\{\ell \mid \exists q\gamma \in \mathsf{INC} : pX^m Z (\hookrightarrow)^\ell q\gamma\}$; we set $\min \emptyset = \infty$.

Since $pX^m Z \sim_m p$ (by induction on $m$), and thus $pX^m Z \in \mathsf{INC}$ implies $m < k$, we can surely construct $\mathsf{INC}$ for a given pOCA in polynomial space.

▶ **Proposition 10.**

***1.*** *If $pX^m Z \sim qX^n Z$ then $\mathsf{dist}(pX^m Z) = \mathsf{dist}(qX^n Z)$.*

***2.*** *If $\mathsf{dist}(pX^m Z) = \mathsf{dist}(qX^n Z) = \infty$ then $pX^m Z \sim qX^n Z$ iff $pX^m Z \sim_k qX^n Z$.*

The proof is the same as in the non-probabilistic case. (Point 1 is obvious. For Point 2 we verify that the set $\{ (q_1 X^{n_1} Z, q_2 X^{n_2} Z) \mid q_1 X^{n_1} Z \sim_k q_2 X^{n_2} Z$ and $\mathsf{dist}(q_1 X^{n_1} Z) = \mathsf{dist}(q_2 X^{n_2} Z) = \infty$ is a bisimulation.)

Consider a shortest path from $pX^m Z$ to $\mathsf{INC}$ (for large $m$). It is not hard to prove (as in [2, Lemma 10]) that such a path can be based on iterating a simple counter-decreasing cycle (of length $\leq k$), possibly preceded by a polynomial prefix and followed by a polynomial suffix. So (finite) $\mathsf{dist}(pX^m Z)$ can be always expressed by the use of linear functions $\frac{\ell}{e} m + b$ where $\ell, e \leq k$ are the length and the decreasing effect of a simple cycle and $b$ is bounded by a polynomial in $k$. It follows that if we have $\mathsf{dist}(pX^m Z) = \mathsf{dist}(qX^n Z) < \infty$, then $n = \frac{\ell_1 e_2}{e_1 \ell_2} m + b'$, which shows that $(m, n, (p, q))$ lies in one of the above mentioned belts, or in the initial space when $m, n$ are small.

As a consequence, in the background points $(m, n, (p, q))$ we have either $\mathsf{dist}(pX^m Z) = \mathsf{dist}(qX^n Z) = \infty$, and $\chi_\sim(m, n, (p, q)) = 1$ if and only if $pX^m Z \sim_k qX^n Z$, or $\mathsf{dist}(pX^m Z) \neq \mathsf{dist}(qX^n Z)$ (and thus $\chi_\sim(m, n, (p, q)) = 0$). So we can easily compute $\chi_\sim$ for any background point in polynomial space.

The above mentioned shortest paths to $\mathsf{INC}$ also show that if we choose $\psi = k!$ (so $\psi = O(2^{k \log k})$) then we have $pX^m Z \rightarrow^* \mathsf{INC}$ if and only if $pX^{(m+\psi)} Z \rightarrow^* \mathsf{INC}$ (for $m$ larger than some polynomial bound), since the counter-effect of each simple cycle divides $\psi$. Hence $\psi$ is a background period as mentioned above.

A nondeterministic algorithm, verifying that $p_0 X^{m_0} Z \sim q_0 X^{n_0} Z$ for $(m_0, n_0, (p_0, q_0))$ in the initial or belt-space, is based on "moving a vertical window of width 3" (as depicted in Fig. 3(a)); in each phase, the window is moved by 1 (to the right), its intersection with the initial and belt space (containing polynomially many points) is computed, a colouring on this intersection is guessed ($\chi_\sim$ is intended) and its (local) consistency is checked (for which also $\chi_\sim$ on the neighbouring background points is computed). More precisely, in the first, i.e. leftmost, window position a colouring in all three (vertical) slices is guessed and the local consistency in the first two slices is checked; after any later shift of the window by one to the right, a colouring in the new (the rightmost) slice is guessed (the guesses in the previous two slices being remembered), and the consistency in the current middle slice is checked. If this is successfully performed for exponentially many steps, after $(m_0, n_0, (p_0, q_0))$ has been coloured with 1, then it is guaranteed that the algorithm could successfully run forever; the pigeonhole principle induces that each belt could be periodically coloured, with an exponential period compatible with the period of the background-border of the belt. Such a successful run of the algorithm, exponential in time but obviously only polynomial in the required space, is thus a witness of $p_0 X^{m_0} Z \sim q_0 X^{n_0} Z$. Since PSPACE=NPSPACE, we have thus sketched a proof of Theorem 6.

It remains to define precisely the consistency of a colouring, guaranteeing that a successful run of the algorithm really witnesses $p_0 X^{m_0} Z \sim q_0 X^{n_0} Z$. (As already mentioned, this is the main change wrt [2].) We use the following particular variant of characterizing (probabilistic) bisimilarity. Given a pLTS $(S, \Sigma, \rightarrow)$, we say that $(s, t)$ is *consistent w.r.t.* a relation $R$ on $S$ (not necessarily an equivalence) if for each $s \xrightarrow{a} d$ there is $t \xrightarrow{a} d'$, and conversely for each $t \xrightarrow{a} d'$ there is $s \xrightarrow{a} d$, such that $d, d'$ are $R'$-equivalent where $R'$ is the least equivalence containing the set $\{(s', t') \mid s \rightarrow s', t \rightarrow t', (s', t') \in R\}$. A *relation $R$ is consistent* if each $(s, t) \in R$ is consistent w.r.t. $R$. The following proposition can be verified along the standard lines.

▶ **Proposition 11.** *$\sim$ is consistent. If $R$ is consistent then $R \subseteq \sim$.*

Our algorithm can surely (locally) check the above defined consistency of the constructed $\chi$ (i.e. of $R_\chi$).

## 4.2 Bisimilarity of pvPDA is in EXPTIME

It is shown in [15, Theorem 3.3] that the bisimilarity problem for (non-probabilistic) vPDA is EXPTIME-complete. We will show that the same holds for pvPDA. First we show the upper bound:

▶ **Theorem 12.** *The bisimilarity problem for pvPDA is in EXPTIME.*

In [15] the upper bound is proved using a reduction to the model-checking problem for (non-visibly) PDA and the modal $\mu$-calculus. The latter problem is in EXPTIME by [16]. This reduction does not apply in the probabilistic case. The reduction from Section 3 cannot be directly applied either, since it incurs an exponential blowup, yielding only a double-exponential algorithm if combined with the result of [16]. Therefore we proceed as follows: First we give a direct proof for (non-probabilistic) vPDA, i.e., we show via a new proof that the bisimilarity problem for vPDA is in EXPTIME. Then we show that the reduction from Section 3 yields a non-probabilistic vPDA that is exponential only in a way that the new algorithm can be made run in single-exponential time: The crucial observation is that the reduction replaces each step in the pvPDA by three steps in the (non-probabilistic) vPDA. An exponential blowup occurs only in intermediate states of the new LTS. Our algorithm allows to deal with those states in a special pre-processing phase. See Appendix B for details.

## 5 Lower Bounds

In this section we show hardness results for pOCA and pvPDA. We start by defining two gadgets, adapted from [5], that will be used for both results. The gadgets are pLTS that allow us to simulate AND and OR gates using probabilistic bisimilarity. We depict the gadgets in Figure 3(b), where we assume that all edges have probability $1/2$ and have the same label. The gadgets satisfy the following propositions (here $s \xrightarrow{a} t_1 \mid t_2$ is a shorthand for $s \xrightarrow{a} d$ where $d(t_1) = d(t_2) = 0.5$).

▶ **Proposition 13.** (AND-gadget) *Suppose* $s, s', t_1, t_1', t_2, t_2'$ *are states in a pLTS such that* $t_1 \not\sim t_2'$ *and the only transitions outgoing from* $s, s'$ *are* $s \xrightarrow{a} t_1 \mid t_2$ *and* $s' \xrightarrow{a} t_1' \mid t_2'$. *Then* $s \sim s'$ *if and only if* $t_1 \sim t_1' \wedge t_2 \sim t_2'$.

▶ **Proposition 14.** (OR-gadget) *Suppose* $s, s', t_1, t_1', t_2, t_2',$ *and* $u_{12}, u_{1'2}, u_{12'}, u_{1'2'}$ *are states in a pLTS. Let the only transitions outgoing from* $s, s', u_{12}, u_{1'2}, u_{12'}, u_{1'2'}$ *be*

$$s \xrightarrow{a} u_{12} \mid u_{1'2'} \, , \, s' \xrightarrow{a} u_{12'} \mid u_{1'2} \, ,$$
$$u_{12} \xrightarrow{a} t_1 \mid t_2 \, , \, u_{1'2'} \xrightarrow{a} t_1' \mid t_2' \, , \, u_{12'} \xrightarrow{a} t_1 \mid t_2' \, , \, u_{1'2} \xrightarrow{a} t_1' \mid t_2 \, .$$

*Then* $s \sim s'$ *if and only if* $t_1 \sim t_1' \vee t_2 \sim t_2'$.

## 5.1 Bisimilarity of pOCA is PSPACE-hard

In this section we prove the following:

▶ **Theorem 15.** *Bisimilarity for pOCA is PSPACE-hard, even for unary (i.e., with only one action) and fully probabilistic pOCA, and for fixed initial configurations of the form* $pXZ, qXZ$.

In combination with Theorem 6 we obtain:

▶ **Corollary 16.** *The bisimilarity problem for pOCA is PSPACE-complete.*

**Proof of Theorem 15.** We use a reduction from the emptiness problem for alternating finite automata with a one-letter alphabet, known to be PSPACE-complete [8, 9]; our reduction resembles the reduction in [15] for (non-probabilistic) visibly one-counter automata.

A *one-letter alphabet alternating finite automaton*, 1L-AFA, is a tuple $A = (Q, \delta, q_0, F)$ where $Q$ is the (finite) set of *states*, $q_0$ is the *initial state*, $F \subseteq Q$ is the set of *accepting states*, and the *transition function* $\delta$ assigns to each $q \in Q$ either $q_1 \wedge q_2$ or $q_1 \vee q_2$, where $q_1, q_2 \in Q$.

We define the predicate $Acc \subseteq Q \times \mathbb{N}$ by induction on the second component (i.e. the length of a one-letter word); $Acc(q, n)$ means "$A$ starting in $q$ accepts $n$": $Acc(q, 0)$ if and only if $q \in F$; $Acc(q, n{+}1)$ if and only if either $\delta(q) = q_1 \wedge q_2$ and we have both $Acc(q_1, n)$ and $Acc(q_2, n)$, or $\delta(q) = q_1 \vee q_2$ and we have $Acc(q_1, n)$ or $Acc(q_2, n)$.

The *emptiness problem for 1L-AFA* asks, given a 1L-AFA $A$, if the set $\{n \mid Acc(q_0, n)\}$ is empty.

We reduce the emptiness of 1L-AFA to our problem. We thus assume a 1L-AFA $(Q, \delta, q_0, F)$, and we construct a pOCA $\Delta$ as follows. $\Delta$ has $2|Q| + 3$ 'basic' states; the set of basic states is $\{p, p', r\} \cup Q \cup Q'$ where $Q' = \{q' \mid q \in Q\}$ is a copy of $Q$ and $r$ is a special dead state. Additional auxiliary states will be added to implement AND- and OR-gadgets. $\Delta$ will have only one input letter, denoted $a$, and will be fully probabilistic.

We aim to achieve $pXZ \sim p'XZ$ if and only if $\{n \mid Acc(q_0, n)\}$ is empty; another property will be that

$$qX^n Z \sim q'X^n Z \text{ if and only if } \neg Acc(q, n). \tag{1}$$

For each $q \in F$ we add a transition $qZ \overset{a}{\hookrightarrow} d$ where $d(r, Z) = 1$, but $qZ$ is dead (i.e., there is no transition $qZ \overset{a}{\hookrightarrow} ..$) if $q \notin F$; $q'Z$ is dead for any $q' \in Q'$. Both $rX$ and $rZ$ are dead as well. Hence (1) is satisfied for $n = 0$. Now we show (1) holds for $n > 0$.

For $q$ with $\delta(q) = q_1 \vee q_2$ we implement an AND-gadget from Figure 3(b) (top) guaranteeing $qX^{n+1}Z \sim q'X^{n+1}Z$ if and only if $q_1 X^n Z \sim q_1' X^n Z$ and $q_2 X^n Z \sim q_2' X^n Z$ (since $\neg Acc(q, n{+}1)$ if and only if $\neg Acc(q_1, n)$ and $\neg Acc(q_2, n)$):

We add rules $qX \to r_1 X \mid r_2 X$ (this is a shorthand for $qX \overset{a}{\hookrightarrow} [r_1 X \mapsto 0.5, r_2 X \mapsto 0.5]$) and $q'X \to r_1'X \mid r_2'X$,
and also $r_1 X \to q_1 \mid s_1 X$, $r_2 X \to q_2 \mid s_2 X$, $r_1'X \to q_1' \mid s_1 X$, $r_2'X \to q_2' \mid s_2 X$,
and $s_1 X \overset{0.5}{\longrightarrow} s_1 X$, $s_1 X \overset{0.5}{\longrightarrow} r$, $s_2 X \overset{0.4}{\longrightarrow} s_2 X$, $s_2 X \overset{0.6}{\longrightarrow} r$. The intermediate states $r_1, r_2, r_1', r_2'$, and $s_1, s_2$ serve to implement the condition $t_1 \not\sim t_2'$ from Proposition 13.

For $q$ with $\delta(q) = q_1 \wedge q_2$ we (easily) implement an OR-gadget from Figure 3(b) (bottom) guaranteeing $qX^{n+1}Z \sim q'X^{n+1}Z$ if and only if $q_1 X^n Z \sim q_1' X^n Z$ or $q_2 X^n Z \sim q_2' X^n Z$ (since $\neg Acc(q, n{+}1)$ if and only if $\neg Acc(q_1, n)$ or $\neg Acc(q_2, n)$).

To finish the construction, we add transitions $pX \overset{a}{\hookrightarrow} d$ where $d(p, X^2) = d(q_0, \varepsilon) = d(r, X) = \frac{1}{3}$ and $p'X \overset{a}{\hookrightarrow} d'$ where $d'(p', X^2) = d(q_0', \varepsilon) = d(r, X) = \frac{1}{3}$; the transitions added before guarantee that $pX^{n+2}Z \not\sim q_0' X^n Z$ and $q_0 X^n Z \not\sim p'X^{n+2}Z$. ◀

## 5.2 Bisimilarity of pvPDA is EXPTIME-hard

In this section we prove the following:

▶ **Theorem 17.** *Bisimilarity for pvPDA is EXPTIME-hard, even for fully probabilistic pvPDA with $|\Sigma_r| = |\Sigma_{int}| = |\Sigma_c| = 1$.*

In combination with Theorem 12 we obtain:

▶ **Corollary 18.** *The bisimilarity problem for pvPDA is EXPTIME-complete.*

It was shown in [15] that bisimilarity for (non-probabilistic) vPDA is EXPTIME-complete. The hardness result there follows by observing that the proof given in [12] for general PDA works in fact even for vPDA. Referring to the conference version of [12], it is commented in [15]: "Though conceptually elegant, the technical details of the reduction are rather tedious." For those reasons we give a full reduction from the problem of determining the winner in a reachability game on pushdown processes. This problem was shown EXPTIME-complete in [16]. Our reduction proves Theorem 17, i.e., for unary and fully probabilistic pvPDA, and at the same time provides a concise proof for (non-probabilistic) vPDA.

**Proof of Theorem 17.** Let $\Delta = (Q, \Gamma, \{a\}, \hookrightarrow)$ be a unary non-probabilistic PDA with a control state partition $Q = Q_0 \cup Q_1$ and an initial configuration $p_0 X_0$. We call a configuration $pX\alpha$ *dead* if it has no successor configuration, i.e., if $\Delta$ does not have a rule with $pX$ on the left-hand side. Consider the following game between Player 0 and Player 1 on the LTS $\mathcal{S}(\Delta)$ induced by $\Delta$: The game starts in $p_0 X_0$. Whenever the game is in a configuration $p\alpha$ with $p \in Q_i$ (where $i \in \{0, 1\}$), Player $i$ chooses a successor configuration of $p\alpha$ in $\mathcal{S}(\Delta)$. The goal of Player 1 is to reach a dead configuration; the goal of Player 0 is to avoid that. It is shown in [16, pp. 261–262] that determining the winner in that game is EXPTIME-hard.

W.l.o.g. we can assume that each configuration has at most two successor configurations, and that no configuration with empty stack is reachable. We construct a fully probabilistic pvPDA $\bar\Delta = (\bar Q, \Gamma, \{a_r, a_{int}, a_c\}, \circ\!\rightarrow)$ such that the configurations $p_0 X_0$ and $p'_0 X_0$ of $\bar\Delta$ are bisimilar if and only if Player 0 can win the game. For each control state $p \in Q$ the set $\bar Q$ includes $p$ and a copy $p'$.

For each $pX \in Q \times \Gamma$, if $pX$ is dead in $\Delta$, we add a rule $pX \xrightarrow{a_{int}, 1} pX$ in $\bar\Delta$, and a rule $p'X \xrightarrow{a_{int}, 1} zX$ where $z \in \bar Q$ is a special control state not occurring on any left-hand side. This ensures that if $pX$ is dead in $\Delta$ (and hence Player 1 wins), then we have $pX \not\sim p'X$ in $\bar\Delta$.

For each $pX \in Q \times \Gamma$ that has in $\Delta$ a single successor configuration $q\alpha$, we add rules $pX \xrightarrow{a, 1} q\alpha$ and $p'X \xrightarrow{a, 1} q'\alpha$, where $a = a_r, a_{int}, a_c$ if $|\alpha| = 0, 1, 2$, respectively.

For each $pX \in Q \times \Gamma$ that has in $\Delta$ two successor configurations, let $p_1 \alpha_1$ and $p_2 \alpha_2$ denote the successor configurations. W.l.o.g. we can assume that $\alpha_1 = X_1 \in \Gamma$ and $\alpha_2 = X_2 \in \Gamma$.

- If $p \in Q_0$ we implement an OR-gadget from Figure 3(b): let $(p_1 X_1 p_2 X_2), (p'_1 X_1 p'_2 X_2), (p_1 X_1 p'_2 X_2), (p'_1 X_1 p_2 X_2) \in \bar Q$ be fresh control states, and add rules $pX \circ\!\rightarrow (p_1 X_1 p_2 X_2)X \mid (p'_1 X_1 p'_2 X_2)X$ (this is a short-hand for $pX \xrightarrow{a_{int}, 0.5} (p_1 X_1 p_2 X_2)X$ and $pX \xrightarrow{a_{int}, 0.5} (p'_1 X_1 p'_2 X_2)X$) and $p'X \circ\!\rightarrow (p_1 X_1 p'_2 X_2)X \mid (p'_1 X_1 p_2 X_2)X$ as well as $(p_1 X_1 p_2 X_2)X \circ\!\rightarrow p_1 X_1 \mid p_2 X_2$ and $(p'_1 X_1 p'_2 X_2)X \circ\!\rightarrow p'_1 X_1 \mid p'_2 X_2$ and $(p_1 X_1 p'_2 X_2)X \circ\!\rightarrow p_1 X_1 \mid p'_2 X_2$ and $(p'_1 X_1 p_2 X_2)X \circ\!\rightarrow p'_1 X_1 \mid p_2 X_2$.

- If $p \in Q_0$ we implement an AND-gadget from Figure 3(b): let $(p_1 X_1), (p'_1 X_1), (p_2 X_2), (p'_2 X_2) \in \bar Q$ be fresh control states, and add rules $pX \circ\!\rightarrow (p_1 X_1)X \mid (p_2 X_2)X$ and $p'X \circ\!\rightarrow (p'_1 X_1)X \mid (p'_2 X_2)X$ as well as $(p_1 X_1)X \xrightarrow{a_{int}, 1} p_1 X_1$ and $(p'_1 X_1)X \xrightarrow{a_{int}, 1} p'_1 X_1$ and $(p_2 X_2)X \circ\!\rightarrow p_2 X_2 \mid zX$ and $(p'_2 X_2)X \circ\!\rightarrow p'_2 X_2 \mid zX$. Here, the transitions to $zX$ serve to implement the condition $t_1 \not\sim t'_2$ from Proposition 13.

An induction argument now easily establishes that $p_0 X_0 \sim p'_0 X_0$ holds in $\bar\Delta$ if and only if Player 0 can win the game in $\Delta$.

We remark that exactly the same reduction works for non-probabilistic vPDA, if the probabilistic branching is replaced by nondeterministic branching. ◀

## References

**1**  C. Baier. Polynomial time algorithms for testing probabilistic bisimulation and simulation. In *CAV*, pages 50–61, 1996.

**2**  S. Böhm, S. Göller, and P. Jančar. Bisimilarity of one-counter processes is PSPACE-complete. In *CONCUR*, volume 6269 of *LNCS*, pages 177–191, 2010.

**3**  T. Brázdil, A. Kučera, and O. Stražovský. Deciding probabilistic bisimilarity over infinite-state probabilistic systems. *Acta Inf.*, 45(2):131–154, 2008.

**4**  O. Burkart, D. Caucal, F. Moller, and B. Steffen. Verification on infinite structures. In J.A. Bergstra, A. Ponse, and S.A. Smolka, editors, *Handbook of Process Algebra*, pages 545–623. North-Holland, 2001.

**5**  D. Chen, F. van Breugel, and J. Worrell. On the complexity of computing probabilistic bisimilarity. In *FoSSaCS*, volume 7213 of *LNCS*, pages 437–451, 2012.

**6**  K. Etessami, D. Wojtczak, and M. Yannakakis. Quasi-birth-death processes, tree-like QBDs, probabilistic 1-counter automata, and pushdown systems. *Perform. Eval.*, 67(9):837–857, 2010.

**7**  H. Fu and J.-P. Katoen. Deciding probabilistic simulation between probabilistic pushdown automata and finite-state systems. In *FSTTCS*, pages 445–456, 2011.

**8**  M. Holzer. On emptiness and counting for alternating finite automata. In *Developments in Language Theory*, pages 88–97, 1995.

**9**  P. Jančar and Z. Sawa. A note on emptiness for alternating finite automata with a one-letter alphabet. *Inf. Process. Lett.*, 104(5):164–167, 2007.

**10**  P. Jančar. Bisimilarity on Basic Process Algebra is in 2-ExpTime (an explicit proof). *CoRR*, abs/1207.2479, 2012.

**11**  S. Kiefer. BPA bisimilarity is EXPTIME-hard. *CoRR*, abs/1205.7041, 2012.

**12**  A. Kučera and R. Mayr. On the complexity of checking semantic equivalences between pushdown processes and finite-state processes. *Information and Computation*, 208(7):772–796, 2010.

**13**  R. Segala and N. A. Lynch. Probabilistic simulations for probabilistic processes. In *CONCUR*, volume 836 of *LNCS*, pages 481–496. Springer, 1994.

**14**  G. Sénizergues. The bisimulation problem for equational graphs of finite out-degree. *SIAM J. Comput.*, 34(5):1025–1106, 2005.

**15**  J. Srba. Beyond language equivalence on visibly pushdown automata. *Logical Methods in Computer Science*, 5(1):2, 2009.

**16**  I. Walukiewicz. Pushdown processes: Games and model-checking. *Information and Computation*, 164(2):234–263, 2001.

In this section we present proofs of some claims from Section 3

## A.1 Proof of Lemma 3

Lemma 3 follows immediately from the following lemma.

▶ **Lemma 19.** *For all configurations $qX\gamma$ and $rY\delta$ of $\Delta$ we have $qX\gamma \sim_n rY\delta$ in $\Delta$ if and only if $qX\gamma \sim'_{3 \cdot n} rY\delta$ in $\Delta'$.*

**Proof.** In what follows, given a distribution $d = [\alpha_1 \mapsto x_1, \ldots, \alpha_n \mapsto x_n]$, we use $d^w$ to denote the distribution $[\alpha_1 w \mapsto x_1, \ldots, \alpha_n w \mapsto x_n]$. Also, we use $\sim'$ to denote the relation $\sim$ of $\Delta'$, to distinguish it from the relation $\sim$ of $\Delta$.

Let us start with the direction $\Rightarrow$ of the lemma. For $n = 0$ the claim obviously holds. Assume it holds for all numbers lower than $n$. Let $qX\gamma$ and $rY\delta$ be configurations of $\Delta$ such that $qX\gamma \sim_n rY\delta$. W.l.o.g. let us pick any transition $qX \overset{a}{\multimap} q\langle d_1\rangle$ where $d_1 = [q_1\beta_1 \mapsto x_1, \ldots, q_n\beta_n \mapsto x_n]$. There must be a transition $rY \overset{a}{\multimap} r\langle d_2\rangle$ where $d_2 = [r_1\beta_1 \mapsto y_1, \ldots, r_m\beta_m \mapsto y_m]$ such that $d_1^\gamma$ and $d_2^\delta$ are $\sim_{n-1}$-equivalent. Let $q\langle d_1\rangle \overset{x}{\multimap} q\langle\{q_{i_1}\alpha_{i_1}, \ldots q_{i_k}\alpha_{i_k}\}\rangle$ be an arbitrary rule with $q\langle d_1\rangle$ on the left hand side (the case of $r\langle d_2\rangle$ is similar). For the set $T = \{q_{i_1}\alpha_{i_1}\gamma, \ldots q_{i_k}\alpha_{i_k}\gamma\}$, we have $x \leq d_1(T)$ and there must be a set $T' = \{r_{j_1}\beta_{j_1}\delta, \ldots, r_{j_\ell}\beta_{j_\ell}\delta\}$ satisfying the conditions of Lemma 2 such that $d_2(T') \geq d_1(T) \geq x$. Hence there is an action $r\langle d_2\rangle \overset{x}{\multimap} r\langle\{r_{j_1}\beta_{j_1}, \ldots r_{j_\ell}\beta_{j_\ell}\}\rangle$. Because $T$ and $T'$ were chosen to satisfy the conditions of Lemma 2, for any $r\langle\{r_{j_1}\beta_{j_1}, \ldots r_{j_\ell}\beta_{j_\ell}\}\rangle\delta \overset{b}{\to} r_j\beta_j\delta$ (these are the only actions available) there is an action $q\langle\{q_{i_1}\alpha_{i_1}, \ldots q_{i_k}\alpha_{i_k}\}\rangle \overset{b}{\to} q_i\alpha_i\gamma$ such that $r_j\beta_j\delta \sim_n q_i\alpha_i\gamma$, and vice versa.

Now let us analyse $\Leftarrow$. For $n = 0$ the claim obviously holds. Assume it holds for all numbers lower than $n$. Let $qX\gamma$ and $rY\delta$ be configurations of $\Delta$ such that $qX\gamma \sim'_{3 \cdot n} rY\delta$ in $\Delta'$. Let $qX\gamma \overset{a}{\multimap} d$ be arbitrary rule, then there is a transition $qX\gamma \overset{a}{\to} q\langle d\rangle\gamma$ in $\Delta'$ and $rY\delta \overset{a}{\to} r\langle d'\rangle\delta$ such that $q\langle d\rangle\gamma \sim'_{3 \cdot n-1} r\langle d'\rangle\delta$. There is also a rule $rY\delta \overset{a}{\multimap} d'$, so to finish the proof it suffices to see that $d^\gamma$ and $(d')^\delta$ are $\sim_{n-1}$-equivalent. Let $T = \{q_{i_1}\alpha_{i_1}\gamma, \ldots, q_{i_k}\alpha_{i_k}\gamma\} \subseteq support(d^\gamma)$ be arbitrary (for the subsets of $support((d')^\gamma)$ the proof is similar), and let $x = d^\gamma(T)$. There is a transition $q\langle d\rangle\gamma \overset{x}{\to} q\langle\{q_{i_1}\alpha_{i_1}, \ldots, q_{i_k}\alpha_{i_k}\}\rangle\gamma$, and hence a transition $r\langle d'\rangle\delta \overset{x}{\to} r\langle\{r_{j_1}\beta_{j_1}, \ldots, r_{j_\ell}\beta_{j_k}\}\rangle\delta$ such that

$$q\langle\{q_{i_1}\alpha_{i_1}, \ldots, q_{i_k}\alpha_{i_k}\}\rangle\gamma \sim'_{3 \cdot n-2} r\langle\{r_{j_1}\beta_{j_1}, \ldots, r_{j_\ell}\beta_{j_k}\}\rangle\delta \tag{2}$$

We put $T' = \{r_{j_1}\beta_{j_1}\delta, \ldots, r_{j_\ell}\beta_{j_\ell}\delta\}$. We show that $T$ and $T'$ satisfy the conditions from Lemma 2 for the relation $\sim_{n-1}$. First, due to the construction of rules available under $x$ we have $d^\gamma(T) \leq (d')^\delta(T')$. Further, for an arbitrary element $q_i\alpha_i\gamma$ there is a transition $q\langle\{q_{i_1}\alpha_{i_1}, \ldots, q_{i_k}\alpha_{i_k}\}\rangle\gamma \overset{b}{\to} q_i\alpha_i\gamma$, and so due to Equation 2 there must be a transition $r\langle\{r_{j_1}\beta_{j_1}, \ldots, r_{j_\ell}\beta_{j_k}\}\rangle\delta \overset{b}{\to} r_j\beta_j\delta$ such that $q_i\alpha_i\gamma \sim'_{3n-3} r_j\beta_j\delta$. Using the induction hypothesis we get $q_i\alpha_i\gamma \sim_{n-1} r_j\beta_j\delta$, which finishes the proof. ◀

## A.2 Analysis of the size of $\Delta'$

Let us analyse the size of $\Delta'$. Let $|\varrho|$ be the number of rules of $\Delta$, and let $m$ be the maximal size of the support of a distribution assigned by some rule of $\Delta$. The size of $|\Gamma'|$ is at most $|\Gamma| + |\varrho| + |\varrho| \cdot 2^m$, the size of $\Sigma'$ is at most $|\Sigma| + |R| + 1$, and the number of rules (in $\multimap$) under an action $a \in \Sigma$ is at most $|\varrho|$, under an action $x \in W$ it is at most $|\varrho| \cdot 2^m$, where

$|W| \leq |\varrho| \cdot 2^m$. The number of rules under the action $\#$ is at most $|\varrho| \cdot 2^m \cdot m$. Hence the size of $\Delta'$ is exponential in the size of $\Delta$, but polynomial when the size of the support of distributions assigned by rules is fixed. Obviously, the construction can be done in time exponential (or polynomial, respectively) in the size of $\Delta$.

## B    Proofs Omitted from Section 4

### B.1    Proof of Theorem 12

We prove the following theorem from the main body of the paper:

**Theorem 12.** *The bisimilarity problem for pvPDA is in EXPTIME.*

**Proof.** The proof is structured as follows. First we show that bisimilarity for (non-probabilistic) vPDA is in EXPTIME, thus reproving a result from [15] via a different method. Then we show that, although the reduction from Section 3 yields an exponential blow-up in translating from pvPDA to vPDA, our new algorithm for deciding bisimilarity on vPDA can still be made to run in single-exponential time in the size of the original pvPDA.

Let $p_0\alpha_0$ and $q_0\beta_0$ be the given initial configurations. W.l.o.g. we assume that $\alpha_0 = X_0 \in \Gamma$ and $\beta_0 = Y_0\beta'$ with $Y_0 \in \Gamma$ and $\beta' \in \Gamma^*$. Recall that bisimulation in a labelled transition system can be naturally characterised by a *bisimulation game* between two players, Attacker and Defender. Two states in a labelled transition system are bisimilar if and only if Defender has a winning strategy, see e.g. [15].

We define some notation. For relations $R \subseteq U \times 2^V$ and $S \subseteq V \times 2^W$, we define $(R \bullet S) \subseteq U \times 2^W$ by $R \circ S{\uparrow}$, where $S{\uparrow} := \{(\{v_1, \ldots, v_k\}, \bigcup_{i=1}^{k} A_i) \mid k \geq 0 \,\wedge\, (v_i, A_i) \in S\} \subseteq 2^V \times 2^W$ and $\circ$ stands for the join of two relations. Note that $\emptyset\, S{\uparrow}\, \emptyset$, hence $u\, R\, \emptyset$ implies $u\, (R \bullet S)\, \emptyset$. To avoid notational clutter in the following, if $C$ and $D$ are sets with $c \in C$ and $d \in D$, we often write $CD$ instead of $C \times D$ and $cd$ instead of $(c, d)$.

For finite sets of configurations $C, C' \subseteq Q\Gamma^*$ we call a relation $F \subseteq CC \times 2^{C'C'}$ a $(C, C')$-*forcing relation* if $cd\, F\, S$ implies that Attacker, starting in $cd$, can play so that he either wins or reaches a configuration in $S$ (Defender may choose which configuration in $S$). If $F$ is a $(C, C')$-forcing relation, then

$$F_{/\Gamma} := \{(cXdY, \{c'_1Xd'_1Y, \ldots, c'_kXd'_kY\}) \mid X, Y \in \Gamma \,\wedge\, (cd, \{c'_1d'_1, \ldots, c'_kd'_k\}) \in F\}$$

is a $(C\Gamma, C'\Gamma)$-forcing relation. If $F$ is a $(C, C')$-forcing relation and $F'$ is a $(C', C'')$-forcing relation, then $F \bullet F'$ is a $(C, C'')$-forcing relation. The union of $(C, C')$-forcing relations is a $(C, C')$-forcing relation, so there is a largest $(C, C')$-forcing relation. We have that $p_0X_0 \not\sim q_0Y_0\beta'$ holds if and only

$$(p_0X_0q_0Y_0)\, \hat{F}\, \{pq \in QQ \mid q\beta' \text{ has an outgoing transition}\}$$

holds for the largest $(Q\Gamma, Q)$-forcing relation $\hat{F}$. (In particular, if $\beta' = \varepsilon$, then Attacker wins if and only if $(p_0X_0q_0Y_0)\, \hat{F}\, \emptyset$ holds.) Hence it suffices to compute $\hat{F}$ in exponential time.

For each $a \in \Sigma_c$ we define a "local" $(Q\Gamma, Q\Gamma\Gamma)$-forcing relation $[a]$ by

$$(pXqY)\, [a]\, A \iff \exists\, pX \xrightarrow{a} p'X'X'' : A \supseteq \{p'X'X''q'Y'Y'' \mid qY \xrightarrow{a} q'Y'Y''\}$$
$$\vee\, \exists\, qY \xrightarrow{a} q'Y'Y'' : A \supseteq \{p'X'X''q'Y'Y'' \mid pX \xrightarrow{a} p'X'X''\}.$$

For $a \in \Sigma_{int}$ and $a \in \Sigma_r$ we analogously define local $(Q\Gamma, Q\Gamma)$- and $(Q\Gamma, Q)$-forcing relations $[a]$, respectively. Those forcing relations can be computed in exponential time. Let $\bar{F}$ be

the least solution of the following equation system:

$$F = \bigcup_{a \in \Sigma_r} [a] \quad \cup \quad \bigcup_{a \in \Sigma_{int}} [a] \bullet F \quad \cup \quad \bigcup_{a \in \Sigma_c} [a] \bullet F_{/\Gamma} \bullet F \ .$$

The least fixed point $\bar{F}$ can be computed by a simple Kleene iteration starting from $F = \emptyset$. The iteration terminates after at most $|Q\Gamma Q\Gamma \times 2^{QQ}|$ rounds, each of which takes at most exponential time. It is not hard to see that $\bar{F}$ is the largest $(Q\Gamma, Q)$-forcing relation. It follows that bisimilarity for (non-probabilistic) vPDA can be decided in exponential time.

Now we consider a (probabilistic) pvPDA $\Delta = (Q, \Gamma, \Sigma, \hookrightarrow)$ with action partition $\Sigma = \Sigma_r \cup \Sigma_{int} \cup \Sigma_c$. We use essentially the reduction from Section 3 to compute a (non-probabilistic) vPDA $\Delta' = (Q, \Gamma', \Sigma', \circ\!\!\rightarrow)$, but we need to adapt it slightly to preserve "visibly-ness": Instead of the action $\#$ we need three actions $\#_r \in \Sigma'_r$ and $\#_{int} \in \Sigma'_{int}$ and $\#_c \in \Sigma'_c$ in $\Delta'$. This change does not affect the correctness of the reduction. Observe that $\Sigma'_r = \{\#_r\}$ and $\Sigma'_c = \{\#_c\}$ and $\Sigma'_{int} = \Sigma \cup W \cup \{\#_{int}\}$. For each $a \in \Sigma_c$ we define a local $(Q\Gamma, Q\Gamma\Gamma)$-forcing relation $[a]$ in $\Delta'$ by

$$[a] := [a]' \bullet \left( \bigcup_{w \in W} [w]' \right) \bullet [\#_c]' \ ,$$

where $[\cdot]'$ refers to the local forcing relation $[\cdot]$ defined above, where the definition is applied to the (non-probabilistic) vPDA $\Delta'$. For $a \in \Sigma_{int}$ and $a \in \Sigma_r$ we analogously define local $(Q\Gamma, Q\Gamma)$- and $(Q\Gamma, Q)$-forcing relations $[a]$, respectively. The fact that these are valid forcing relations in $\Delta'$ follows from the structure of the reduction, where each transition is mapped to three consecutive transitions in $\Delta'$.

▶ **Lemma 20.** *For all $a \in \Sigma$, the relation $[a]$ can be computed in exponential time.*

**Proof of the lemma.** We assume $a \in \Sigma_c$; the other cases are similar. It suffices to show that, given $pXqY \in Q\Gamma Q\Gamma$ and $A \subseteq Q\Gamma\Gamma Q\Gamma\Gamma$, we can check in exponential time whether $pXqY \ [a] \ A$ holds. To show this we give an alternating PSPACE algorithm that checks whether $pXqY \ [a] \ A$ holds. Then the lemma follows from APSPACE = EXPTIME. We formulate the APSPACE algorithm in terms of an existential player (corresponding to Attacker) and a universal player (corresponding to Defender):

input: $a \in \Sigma_c$ and $pXqY \in Q\Gamma Q\Gamma$ and $A \subseteq Q\Gamma\Gamma Q\Gamma\Gamma$
return: whether $pXqY\ [a]\ A$ holds
    ex. player: choose either:
        ex. player: choose $d$ s.t. $pX \overset{a}{\multimap\!\!\to} p\langle d\rangle$
        un. player: choose $e$ s.t. $qY \overset{a}{\multimap\!\!\to} q\langle e\rangle$
    or:
        ex. player: choose $e$ s.t. $qY \overset{a}{\multimap\!\!\to} q\langle e\rangle$
        un. player: choose $d$ s.t. $pX \overset{a}{\multimap\!\!\to} p\langle d\rangle$
    ex. player: choose either:
        ex. player: choose $w, T$ s.t. $p\langle d\rangle \overset{w}{\multimap\!\!\to} p\langle T\rangle$
        un. player: choose $U$ s.t. $q\langle e\rangle \overset{w}{\multimap\!\!\to} q\langle U\rangle$
    or:
        ex. player: choose $w, U$ s.t. $q\langle e\rangle \overset{w}{\multimap\!\!\to} q\langle U\rangle$
        un. player: choose $T$ s.t. $p\langle d\rangle \overset{w}{\multimap\!\!\to} p\langle T\rangle$
    ex. player: choose either:
        ex. player: choose $p'X'X''$ s.t. $p\langle T\rangle \overset{\#_c}{\multimap\!\!\to} p'X'X''$
        un. player: choose $q'Y'Y''$ s.t. $q\langle U\rangle \overset{\#_c}{\multimap\!\!\to} q'Y'Y''$
    or:
        ex. player: choose $q'Y'Y''$ s.t. $q\langle U\rangle \overset{\#_c}{\multimap\!\!\to} q'Y'Y''$
        un. player: choose $p'X'X''$ s.t. $p\langle T\rangle \overset{\#_c}{\multimap\!\!\to} p'X'X''$
    return whether $p'X'X''q'Y'Y'' \in A$ holds

◀

We can compute the largest $(Q\Gamma, Q)$-forcing relation $\hat{F}$ as above, i.e., by solving the equation system

$$F = \bigcup_{a \in \Sigma_r} [a] \quad \cup \quad \bigcup_{a \in \Sigma_{int}} [a] \bullet F \quad \cup \quad \bigcup_{a \in \Sigma_c} [a] \bullet F_{/\Gamma} \bullet F$$

using simple Kleene iteration. As above, the iteration terminates after at most $|Q\Gamma Q\Gamma \times 2^{QQ}|$ rounds, each of which takes at most exponential time. This completes the proof. ◀