Logical Methods in Computer Science
Vol. 14(4:13)2018, pp. 1–25
https://lmcs.episciences.org/
Submitted Nov. 18, 2017
Published Nov. 15, 2018

GAME CHARACTERIZATION OF PROBABILISTIC BISIMILARITY, AND APPLICATIONS TO PUSHDOWN AUTOMATA

VOJTĚCH FOREJT [1], PETR JANČAR [2], STEFAN KIEFER [3], AND JAMES WORRELL [3]

[1] Diffblue Ltd
e-mail address: forejtv@gmail.com

[2] Dept of Computer Science, Faculty of Science, Palacký Univ., Olomouc, Czechia
e-mail address: petr.jancar@upol.cz

[3] Dept of Computer Science, University of Oxford, UK
e-mail address: stekie@cs.ox.ac.uk
e-mail address: James.Worrell@cs.ox.ac.uk

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 epsilon-transitions). We first show a general characterization of probabilistic bisimilarity in terms of two-player games, which naturally reduces checking bisimilarity of probabilistic labelled transition systems to checking bisimilarity of standard (non-deterministic) labelled transition systems. This reduction can be easily implemented in the framework of pPDA, allowing to use known results for standard (non-probabilistic) PDA and their subclasses. A direct use of the reduction incurs an exponential increase of complexity, which does not matter in deriving decidability of bisimilarity for pPDA due to the non-elementary complexity of the problem. In the cases of probabilistic one-counter automata (pOCA), of probabilistic visibly pushdown automata (pvPDA), and of probabilistic basic process algebras (i.e., single-state pPDA) we show that an implicit use of the reduction can avoid the complexity increase; we thus get PSPACE, EXPTIME, and 2-EXPTIME upper bounds, respectively, like for the respective non-probabilistic versions. The bisimilarity problems for OCA and vPDA are known to have matching lower bounds (thus being PSPACE-complete and EXPTIME-complete, respectively); we show that these lower bounds also hold for fully probabilistic versions that do not use non-determinism.

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 [BCMS01] for an overview.

Key words and phrases: probabilistic bisimulation equivalence, pushdown automata.

LOGICAL METHODS IN COMPUTER SCIENCE
DOI:10.23638/LMCS-14(4:13)2018

We mention in particular the landmark result showing the decidability of bisimilarity for pushdown automata [Sén05].

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 generalize classical pushdown automata without ε-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 [BCMS01, Srb09]. 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* [EWY10]. Probabilistic BPA seems to have been introduced in [BKS08].

Probabilistic finite-state automata are well understood, including the complexity of bisimilarity [Bai96, BEM00, CvBW12]. Probabilistic pushdown automata, or the equivalent model of recursive Markov chains, have been also studied (we can name [EKM06] and [EY09] among the first respective journal papers) but there are relatively few works on equivalence of infinite-state probabilistic systems. Bisimilarity of probabilistic BPA was shown decidable in [BKS08], but without any complexity bound. In [FK11] probabilistic simulation between pPDA and finite state systems was studied.

We also note that in this paper we consider only systems without ε-transitions; in this context bisimilarity is sometimes also called *strong bisimilarity*. In the literature there are various notions of behavioural equivalences for systems with (silent) ε-transitions, in particular *weak bisimilarity* and *branching bisimilarity*. Such notions have also been studied in the context of probabilistic systems: see [BH97, PLS00] for weak bisimilarity, see [AW06] for branching bisimilarity, and see [ZYS$^+$18] and the references therein for more recent variants.

Our main concern here is to extend the known algorithms for strong bisimilarity on the above mentioned subclasses of pushdown automata to their probabilistic extensions. In the case of weak bisimilarity, the known results are rather negative already for non-probabilistic systems: we can recall the undecidability result for one-counter automata (OCA) [May03], and other results surveyed in (the updated online version of) [Srb04].

Our contribution. We first suggest a characterization of bisimilarity in a probabilistic labelled transition system (pLTS) $\mathcal{L}$ in terms of a two-player game; the game can be viewed as the standard bisimulation game played in a (non-probabilistic) LTS $\mathcal{L}'$ that arises from $\mathcal{L}$ by adding states corresponding to probability distributions and subsets of their supports, while the probabilities of such subsets are "encoded" as new actions. This relatively simple reduction allows us to leverage the rich theory that has been developed for the standard (non-probabilistic) bisimilarity to the probabilistic case. This is in particular straightforward in the case of devices with a control unit or a stack (like pushdown automata and their subclasses); a probabilistic machine M generating a pLTS $\mathcal{L}$ can be easily transformed to a non-probabilistic M' generating the mentioned LTS $\mathcal{L}'$.

A negative feature of the above transformation of M to M' is an exponential increase of the machine size; thus a “blind” use of the reduction easily extends decidability in a standard case to the respective probabilistic case but the upper complexity bound gets increased. A more careful analysis of known algorithms working for standard M is required to show that these algorithms can be modified to be also working for probabilistic M where they, in fact, decide bisimilarity for (exponentially bigger) M' without constructing M' explicitly. Roughly speaking, in the standard bisimulation game the players transform a current pair of states, i.e., of configurations of a standard M, to a new current pair in a *round* of a play. If M is probabilistic, then any standard play for M' starting in a pair of configurations of M visits a pair of configurations of M every three rounds. The mentioned modifications of standard algorithms can be viewed as handling such three rounds as one macro-round.

We now list concrete decidability and complexity results obtained in this paper:

- Using the above-mentioned “blind” reduction together with the result of [Sén05] (for which [Jan14a] gives an alternative proof), we show that bisimilarity for probabilistic pushdown automata is decidable. We do not care about the complexity increase, since the problem is known to be non-elementary [BGKM13] (and even Ackermann-hard [Jan14b] for the model studied in [Sén05]).
- For probabilistic visibly pushdown automata (pvPDA), the reduction immediately yields a 2-EXPTIME upper bound, using the EXPTIME-completeness result in [Srb09] for the standard case; the upper bound in [Srb09] was achieved by a reduction to a result in [Wal01]. Here we give a self-contained short proof that the bisimilarity problem for pvPDA is in EXPTIME; we thus also provide a new proof of the EXPTIME upper bound in the standard case.
- For the class of probabilistic BPA, i.e., pPDA with a single control state, decidability was shown in [BKS08], with no complexity upper bound. Our generic reduction yields a 3-EXPTIME upper bound, using the 2-EXPTIME bound in the standard case (stated in [BCMS01], and explicitly proven in [Jan13]). By a detailed look at the algorithm in [Jan13], we show the modifications that place the problem in 2-EXPTIME also in the probabilistic case. We note that here we have a complexity gap, since only an EXPTIME lower bound for this problem is known, already in the standard case [Kie13].
- The bisimilarity problem for one-counter automata (OCA) is known to be PSPACE-complete. We show here that the upper bound also applies to the probabilistic case (pOCA), by modifying the algorithm described in [BGJ14].
- Finally we show that the completeness results, namely the PSPACE-completeness for pOCA (or OCA) and the EXPTIME-completeness for pvPDA (or vPDA), also hold for *fully* probabilistic OCA and vPDA, respectively. To this aim, we adapt the respective lower bound constructions. (We note that [Kie13] shows that the EXPTIME hardness also holds for fully probabilistic BPA.)

This paper is based on a conference publication [FJKW12]. It has arisen by a substantial rewriting, highlights a crucial idea in a general form, gives complete proofs in a more unified way, and improves a 3-EXPTIME upper bound for pBPA from [FJKW12] to the 2-EXPTIME upper bound, which supports the message discussed in Section 6.

Section 2 contains basic definitions, and Section 3 then shows a game characterization of probabilistic bisimilarity, and a reduction to standard bisimilarity. Section 4 provides the announced complexity upper bounds, while Section 5 shows the lower bounds. Section 6 contains some concluding remarks.

2. Preliminaries

By $\mathbb{N}$ and $\mathbb{Q}$ we denote the sets of nonnegative integers and of rationals, respectively. Given a finite or countable set A, a *probability distribution on A* is a function $d\colon A \to [0,1]\cap\mathbb{Q}$ such that $\sum_{a\in A} d(a) = 1$; the *support* of d is the set $\mathit{support}(d) = \{\, a \in A \mid d(a) > 0 \,\}$. The set of all probability distributions on A is denoted by $\mathcal{D}(A)$. A probability distribution $d \in \mathcal{D}(A)$ is *Dirac* if $d(a) = 1$ for some $a \in A$ (and $d(y) = 0$ for all $y \neq a$). For any set $B \subseteq A$ we define $d(B) = \sum_{a\in B} d(a)$. When there is no confusion, we may write $\sum_{a\in \mathit{support}(d)} d(a)a$ to indicate d. E.g., for d with $d(a) = \frac{1}{3}$ and $d(b) = \frac{2}{3}$ we may write $\frac{1}{3}a + \frac{2}{3}b$; if $d(a) = 1$, then we may write d as $1a$ or simply as a.

2.1. Labelled Transition Systems (pLTSs, fpLTSs, LTSs).

A *probabilistic labelled transition system (pLTS)* is a tuple $\mathcal{L} = (S, \Sigma, \to)$, where S is a finite or countable set of *states*, Σ is a finite *action alphabet*, and $\to\ \subseteq S \times \Sigma \times \mathcal{D}(S)$ is a *transition relation*. Throughout the paper we assume that pLTS are *image-finite*, that is, we assume that for each $s \in S$ and $a \in \Sigma$ there are only finitely many $d \in \mathcal{D}(S)$ such that $(s, a, d) \in\ \to$.

We usually write $s \xrightarrow{a} d$ instead of $(s, a, d) \in\ \to$. By $s \to s'$ we denote that there is a transition $s \xrightarrow{a} d$ with $s' \in \mathit{support}(d)$. (Our definition allows that the set $\{\, s' \mid s \to s' \,\}$ might be infinite for a state s, though such sets are finite in the later applications.) State s' is *reachable from* s if $s \to^* s'$, where $\to^*$ is the reflexive and transitive closure of $\to$.

In general a pLTS combines non-deterministic and probabilistic branching. A pLTS $\mathcal{L} = (S, \Sigma, \to)$ is *fully probabilistic*, an *fpLTS*, if for each pair $s \in S$, $a \in \Sigma$ we have $s \xrightarrow{a} d$ for at most one distribution d. A pLTS $\mathcal{L} = (S, \Sigma, \to)$ is a *standard LTS*, or just an *LTS* for short, if in each $s \xrightarrow{a} d$ the distribution d is Dirac; in this case $s \xrightarrow{a} d$ is instead written as $s \xrightarrow{a} s'$ where $d(s') = 1$.

Let $\mathcal{L} = (S, \Sigma, \to)$ be a pLTS and R be an equivalence relation on S. We say that two distributions $d, d' \in \mathcal{D}(S)$ are *R-equivalent* if $d(E) = d'(E)$ for each R-equivalence class E. We furthermore say that R is a *bisimulation relation* if $s\ R\ t$ (and thus also $t\ R\ s$) implies that for each transition $s \xrightarrow{a} d$ there is a transition $t \xrightarrow{a} d'$ (with the same action a) such that d and d' are R-equivalent. The union of all bisimulation relations on $\mathcal{S}$ is itself a bisimulation relation; this relation is called *bisimulation equivalence* or *bisimilarity* [SL94]; we denote it by $\sim$.

We also use the following inductive characterization of bisimilarity, assuming a pLTS $\mathcal{L} = (S, \Sigma, \to)$. We define a decreasing sequence of equivalence relations $\sim_0\ \supseteq\ \sim_1\ \supseteq\ \sim_2\ \supseteq \cdots$ on S by putting $s \sim_0 t$ for all s, t, and $s \sim_{n+1} t$ if for each transition $s \xrightarrow{a} d$ there is a transition $t \xrightarrow{a} d'$ such that d, d' are $\sim_n$-equivalent (i.e., $d(E) = d'(E)$ for every $\sim_n$-equivalence class E). It is easy to verify that the sequence $\sim_n$ converges to $\sim$, i.e., $\bigcap_{n\in\mathbb{N}} \sim_n\ =\ \sim$; the fact $\bigcap_{n\in\mathbb{N}} \sim_n\ \supseteq\ \sim$ is trivial (since $\sim_n\ \supseteq\ \sim$ for each $n \in \mathbb{N}$), and $\bigcap_{n\in\mathbb{N}} \sim_n\ \subseteq\ \sim$ holds since $\bigcap_{n\in\mathbb{N}} \sim_n$ is a bisimulation due to our image-finiteness assumption on pLTSs (as can be easily checked).

2.2. **Pushdown Automata (pPDA, fpPDA, PDA) and Their Subclasses.** A *probabilistic pushdown automaton* (pPDA) is a tuple $\Delta = (Q, \Gamma, \Sigma, \hookrightarrow)$ where Q is a finite set of (control) *states*, Γ is a finite *stack alphabet*, Σ is a finite *action alphabet*, and $\hookrightarrow \subseteq Q \times \Gamma \times \Sigma \times \mathcal{D}(Q \times \Gamma^{\leq 2})$ is a set of *(transition) rules*; by $\Gamma^{\leq 2}$ we denote the set $\{\varepsilon\} \cup \Gamma \cup \Gamma\Gamma$ where ε denotes the empty string (and $\Gamma\Gamma = \{\, XY \mid X \in \Gamma, Y \in \Gamma \,\}$). A *configuration* of Δ is a pair $(q, \beta) \in Q \times \Gamma^*$, viewed also as a string $q\beta \in Q\Gamma^*$ (where Γ^* is the set of finite words over alphabet Γ). We write $qX \overset{a}{\hookrightarrow} d$ to denote that (q, X, a, d) is a rule (i.e., an element of $\hookrightarrow$). When speaking of the *size* of Δ, we assume that the probabilities in the rules are given as quotients of integers written in binary.

A pPDA $\Delta = (Q, \Gamma, \Sigma, \hookrightarrow)$ generates the pLTS $\mathcal{L}_\Delta = (Q\Gamma^*, \Sigma, \to)$ defined as follows. For each $\beta \in \Gamma^*$, any rule $qX \overset{a}{\hookrightarrow} d$ of Δ induces a transition $qX\beta \overset{a}{\to} d'$ in $\mathcal{L}_\Delta$ where $d' \in \mathcal{D}(Q\Gamma^*)$ satisfies $d'(p\alpha\beta) = d(p\alpha)$ for each $p\alpha \in \mathit{support}(d)$ (hence $p\alpha \in Q\Gamma^{\leq 2}$). We note that all configurations $q\varepsilon$ (with the empty stack) are "dead" (or terminating) states of $\mathcal{L}_\Delta$.

A tuple $qX \in Q\Gamma$ is called a *head*. A pPDA $\Delta = (Q, \Gamma, \Sigma, \hookrightarrow)$ is *fully probabilistic*, an *fpPDA*, if for each head qX and each action $a \in \Sigma$ there is at most one distribution d such that $qX \overset{a}{\hookrightarrow} d$. The pLTS $\mathcal{L}_\Delta$ generated by an fpPDA Δ is thus an fpLTS.

A *standard PDA*, or a *PDA* for short, is a pPDA where all distributions in the rules are Dirac; in this case $\mathcal{L}_\Delta$ is a (standard) LTS.

A *probabilistic basic process algebra (pBPA)* is a pPDA with only one control state. In this case it is natural to omit the control state in configurations.

A *probabilistic visibly pushdown automaton (pvPDA)* is a pPDA $(Q, \Gamma, \Sigma, \hookrightarrow)$ with a partition of the actions $\Sigma = \Sigma_r \cup \Sigma_{int} \cup \Sigma_c$ such that for all $pX \overset{a}{\hookrightarrow} d$ we have: if $a \in \Sigma_r$ then $\mathit{support}(d) \subseteq Q \times \{\varepsilon\}$; if $a \in \Sigma_{int}$ then $\mathit{support}(d) \subseteq Q \times \Gamma$; if $a \in \Sigma_c$ then $\mathit{support}(d) \subseteq Q \times \Gamma\Gamma$.

A *probabilistic one-counter automaton (pOCA)* is a pPDA $\Delta = (Q, \Gamma, \Sigma, \hookrightarrow)$ where $\Gamma = \{I, Z\}$, for each $p\alpha \in Q\Gamma^{\leq 2}$ such that $d(p\alpha) > 0$ for some $qI \overset{a}{\hookrightarrow} d$ we have $\alpha \in \{\varepsilon, I, II\}$, and for each $p\alpha \in Q\Gamma^{\leq 2}$ such that $d(p\alpha) > 0$ for some $qZ \overset{a}{\hookrightarrow} d$ we have $\alpha \in \{Z, IZ\}$. In this case $\mathcal{L}_\Delta$ is restricted to the set of states $q\alpha Z$ where $\alpha \in \{I\}^*$; this set is closed under reachability due to the form of $\hookrightarrow$. In $q\alpha Z$ the length of α is viewed as a non-negative counter value, while Z always and only occurs at the bottom of the stack.

The fully probabilistic and standard versions of the mentioned subclasses of pPDA, i.e., *fpBPA*, *fpvPDA*, *fpOCA*, *BPA*, *vPDA*, *OCA*, are defined analogously as in the general case.

The *bisimilarity problem* for pPDA asks whether two configurations $q_1\alpha_1$ and $q_2\alpha_2$ of a given pPDA Δ are bisimilar when regarded as states of the generated pLTS $\mathcal{L}_\Delta$. We will also consider restrictions of the problem to subclasses of pPDA.

Example 2.1. Consider the fpPDA $\Delta = (\{p, q, r\}, \{X, X', Y, Z\}, \{a\}, \hookrightarrow)$ with the following rules (omitting the unique action a):

$$\begin{aligned} pX &\hookrightarrow 0.5qXX + 0.5p & qX &\hookrightarrow pXX \\ rX &\hookrightarrow 0.3rYX + 0.2rYX' + 0.5r & rY &\hookrightarrow rXX \\ rX' &\hookrightarrow 0.4rYX + 0.1rYX' + 0.5r \end{aligned}$$

The restriction of Δ to the control states p, q and to the stack symbols X, Z yields a pOCA, when X plays the role of I from the definition. The restriction of Δ to the control state r and the stack symbols X, X', Y yields a pBPA. A fragment of the pLTS $\mathcal{L}_\Delta$ is shown in Figure 1.

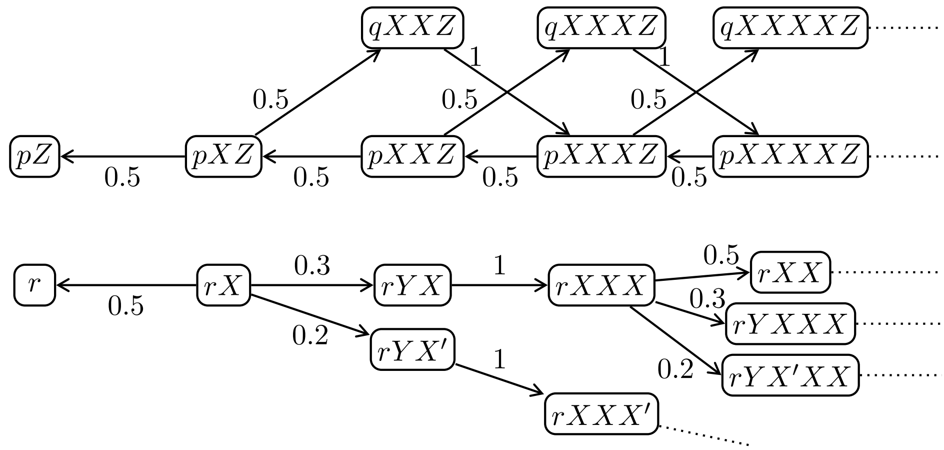


Figure 1: A fragment of $\mathcal{L}_\Delta$ from Example 2.1.

The configurations pXZ and rX are bisimilar, as there is a bisimulation relation with equivalence classes $\{pX^kZ\} \cup \{\, rw \mid w \in \{X, X'\}^k \,\}$ for all $k \geq 0$ and $\{qX^{k+1}Z\} \cup \{\, rYw \mid w \in \{X, X'\}^k \,\}$ for all $k \geq 1$.

3. Game Characterization of Probabilistic Bisimilarity

Bisimilarity in standard LTSs has a natural characterization in terms of two-player games. This game-theoretic characterization is a source of intuition and allows for elegant presentations of certain arguments. The players in the bisimulation game can be called *Attacker* and *Defender*; a play of such a game, in an LTS $\mathcal{L} = (S, \Sigma, \rightarrow)$, yields a sequence of pairs of states, starting from a given initial pair (s, s'). The objective of the game for *Attacker* is a reachability objective. Specifically, pairs of states (s_1, s_2) for which s_1 and s_2 enable different sets of actions are declared as Attacker goals; Attacker aims to reach such a pair, Defender aims to avoid this.

A play arises as follows. Given a current pair (s_1, s_2), the players perform a prescribed protocol that results either in a demonstration that (s_1, s_2) is an Attacker goal, in which case the play finishes, or in creating a new current pair (s'_1, s'_2). In the standard bisimilarity game the protocol is the following: Attacker chooses a transition $s_i \xrightarrow{a} s'_i$ for some $i \in \{1, 2\}$, $a \in \Sigma$, $s'_i \in S$, and Defender responds by choosing $s_{3-i} \xrightarrow{a} s'_{3-i}$ for some $s'_{3-i} \in S$; this yields a pair (s'_1, s'_2). If Defender has no possible response (action a is not enabled in s_{3-i}), then the play finishes since it has been demonstrated that (s_1, s_2) is an Attacker goal. If no action is enabled in any of s_1, s_2, then we can formally put $(s'_1, s'_2) = (s_1, s_2)$, or simply stop the play as a win of Defender.

It is easy to observe that $s \sim_n s'$ iff Attacker cannot force his win in n rounds of the play (i.e., within n runnings of the protocol) when (s, s') is the initial current pair.

In the case of *probabilistic bisimilarity*, i.e., of bisimilarity in probabilistic LTSs, a similar game characterization is not immediately obvious. We suggest the characterization arising by the following modification of the above game, now in a pLTS $\mathcal{L} = (S, \Sigma, \rightarrow)$; we define the protocol, to be performed in a round of the game from a current pair (s_1, s_2), as follows:

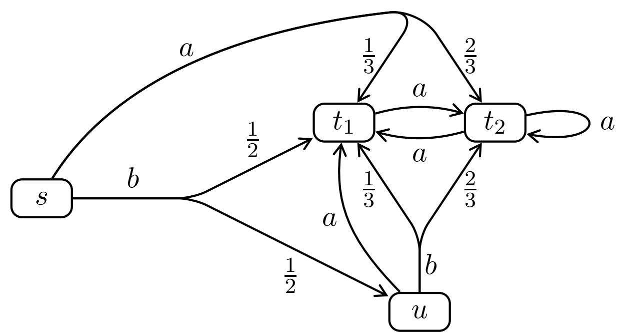


Figure 2: A finite pLTS $\mathcal{L}$ with bisimulation equivalence classes $\{s\}, \{t_1, t_2\}, \{u\}$.

(1) Attacker chooses $s_i \xrightarrow{a} d_i$, for some $i \in \{1,2\}$ and $(s_i, a, d_i) \in \rightarrow$; if this is not possible, the play stops by declaring Defender the winner. Defender chooses $s_{3-i} \xrightarrow{a} d_{3-i}$ for some $(s_{3-i}, a, d_{3-i}) \in \rightarrow$; if not possible, then the play finishes by declaring Attacker the winner (it has been demonstrated that an Attacker goal has been reached).
(2) Attacker chooses a nonempty subset $T_j \subseteq \mathit{support}(d_j)$ for some $j \in \{1,2\}$; Defender chooses some $T_{3-j} \subseteq \mathit{support}(d_{3-j})$ such that $d_{3-j}(T_{3-j}) \geq d_j(T_j)$.
(3) Attacker chooses $s'_k \in T_k$ for some $k \in \{1,2\}$; Defender chooses $s'_{3-k} \in T_{3-k}$.
(4) The pair (s'_1, s'_2) becomes a new current pair.

Example 3.1. Figure 2 shows a finite pLTS $\mathcal{L} = (S, \Sigma, \rightarrow)$. Here is a play that proves $s \not\sim_2 u$. The play starts in (s,u). Attacker chooses $s \xrightarrow{b} d_1$ where $d_1 = \frac{1}{2}u + \frac{1}{2}t_1$. This forces Defender to choose $u \xrightarrow{b} d_2$ where $d_2 = \frac{1}{3}t_1 + \frac{2}{3}t_2$. Now Attacker may choose $\{t_2\} \subseteq \mathit{support}(d_2)$, where $d_2(\{t_2\}) = \frac{2}{3}$. Defender now has to choose $T_1 \subseteq \mathit{support}(d_1) = \{u, t_1\}$ so that $d_1(T_1) \geq \frac{2}{3}$, i.e., she has to choose $T_1 = \{u, t_1\}$. Attacker can now take $u \in T_1$, and Defender can only take t_2. Thus the new pair is (u, t_2). In the next round Attacker chooses $u \xrightarrow{b} d_2$, and Defender cannot respond. That is, (u, t_2) is an Attacker goal, and Attacker has won. Thus $s \not\sim_2 u$.

It might not be obvious in general that $s \sim_n s'$ (in the given pLTS) iff Attacker cannot force a win in n rounds when starting from (s, s'). We prove this formally below; moreover, we intentionally implement the protocol as a standard bisimulation (sub)game (consisting of three rounds), since this makes it easier to lift known results for standard bisimilarity to the case of probabilistic bisimilarity. Hence we transform a given probabilistic LTS $\mathcal{L}$ to a standard LTS $\mathcal{L}'$ that extends the state set of $\mathcal{L}$ by viewing distributions and also nonempty subsets of their supports as explicit states; we will guarantee that a pair of original states will be bisimilar in (the probabilistic) $\mathcal{L}$ iff it is bisimilar in (the standard) $\mathcal{L}'$.

Now we describe the transformation of $\mathcal{L}$ to $\mathcal{L}'$ rigorously. We assume a pLTS $\mathcal{L} = (S, \Sigma, \rightarrow)$ and use the following technical notions:

- a *distribution* $d \in \mathcal{D}(S)$ is *relevant* if $s \xrightarrow{a} d$ for some $s \in S$, $a \in \Sigma$;
- a *nonempty set of states* $T \subseteq S$ is *relevant* if $T \subseteq \mathit{support}(d)$ for some relevant d;
- a *number* ρ is *relevant* if $\rho = d(T)$ for some relevant d and $\emptyset \neq T \subseteq \mathit{support}(d)$ (hence $0 < \rho \leq 1$).

The LTS $\mathcal{L}' = (S', \Sigma', \circ\!\!\rightarrow)$ is defined as follows:

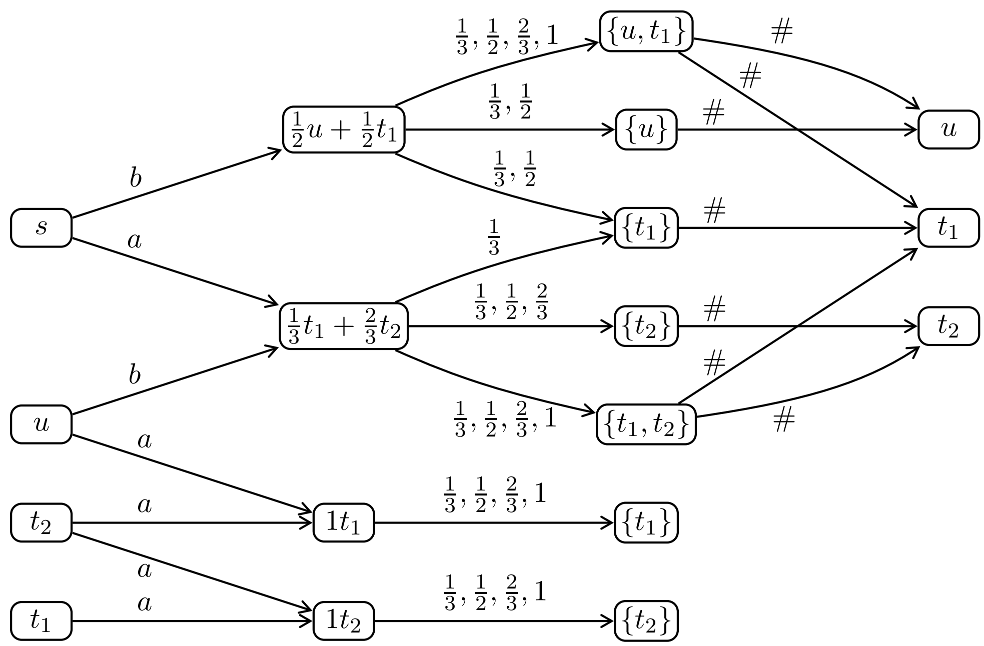


Figure 3: The standard LTS $\mathcal{L}'$ obtained from the probabilistic LTS $\mathcal{L}$ in Figure 2. For better readability, some states appear multiple times but their outgoing transitions are drawn only once.

- $S' = S \cup \{\, d \in \mathcal{D}(S) \mid d \text{ is relevant} \,\} \cup \{\, T \subseteq S \mid T \text{ is relevant} \,\}$. Note that S' is at most countable if each relevant distribution d has finite support.
- $\Sigma' = \Sigma \cup \{\, \rho \in [0,1] \mid \rho \text{ is relevant} \,\} \cup \{\#\}$ where the three parts of the union are pairwise disjoint.
- The relation $\circ\!\!\rightarrow$ is the smallest relation satisfying the following conditions:
  - if $s \xrightarrow{a} d$ (in $\mathcal{L}$), then $s \overset{a}{\circ\!\!\rightarrow} d$ (in $\mathcal{L}'$);
  - if $T \subseteq \mathit{support}(d)$ and $d(T) \geq \rho$ for some relevant d, ρ (hence $T \neq \emptyset$), then $d \overset{\rho}{\circ\!\!\rightarrow} T$;
  - if $s \in T$ (for some relevant T), then $T \overset{\#}{\circ\!\!\rightarrow} s$.

Example 3.2. Figure 3 shows the LTS $\mathcal{L}' = (S', \Sigma', \circ\!\!\rightarrow)$ obtained from $\mathcal{L}$ in Example 3.1 (depicted in Figure 2) according to the construction above. The play from Example 3.1 has a corresponding play in the bisimulation game of $\mathcal{L}'$. We have $s \sim_1 u$ in $\mathcal{L}$ and $s \sim_3 u$ in $\mathcal{L}'$, but $s \not\sim_2 u$ in $\mathcal{L}$ and $s \not\sim_4 u$ (hence $s \not\sim_6 u$) in $\mathcal{L}'$.

The next lemma captures a crucial fact on the relation of $\mathcal{L}$ and $\mathcal{L}'$.

Lemma 3.3. *Given a pLTS $\mathcal{L}$ and the LTS $\mathcal{L}'$ as above, for any states s, s' of $\mathcal{L}$ and any $n \in \mathbb{N}$ we have*

$$s \sim_n s' \text{ in } \mathcal{L} \text{ iff } s \sim_{3n} s' \text{ in } \mathcal{L}'.$$

This also yields that $s \sim s'$ in $\mathcal{L}$ iff $s \sim s'$ in $\mathcal{L}'$.

Proof. We assume $\mathcal{L} = (S, \Sigma, \rightarrow)$ and $\mathcal{L}' = (S', \Sigma', \circ\!\!\rightarrow)$ as above. We recall that $d_1, d_2 \in \mathcal{D}(S)$ are called R-equivalent, for an equivalence R on S, iff $d_1(E) = d_2(E)$ for each R-class E. We use the expression $d_1 \sim_n d_2$ to denote that d_1, d_2 are $\sim_n$-equivalent (for $\sim_n$ in $\mathcal{L}$).

We also say that subsets T_1, T_2 of S are *R-similar* if T_1, T_2 represent the same R-classes, i.e., if the sets $\{ E \mid E$ is an R-class, $E \cap T_1 \neq \emptyset \}$ and $\{ E \mid E$ is an R-class, $E \cap T_2 \neq \emptyset \}$ are the same. By $T_1 \sim_n T_2$ we denote that T_1, T_2 are $\sim_n$-similar (for $\sim_n$ in $\mathcal{L}$).

By $\sim$ we denote bisimilarity in (the pLTS) $\mathcal{L}$, and by $\sim'$ we denote bisimilarity in (the standard LTS) $\mathcal{L}'$. It suffices to show that for each $n \in \mathbb{N}$ we have:

(1) $s_1 \sim_n s_2$ iff $s_1 \sim'_{3n} s_2$.
(2) $d_1 \sim_n d_2$ iff $d_1 \sim'_{3n+2} d_2$.
(3) $T_1 \sim_n T_2$ iff $T_1 \sim'_{3n+1} T_2$.

(By s_i we denote elements of S, by d_i relevant distributions, and by T_i relevant sets.)

We show Claims 1–3 by the following observations:

- Claim 1 trivially holds for $n = 0$.
- If Claim 1 holds for n, then Claim 3 holds for n:

  We observe that $T_1 \sim_n T_2$ iff for each $i \in \{1, 2\}$ and each transition $T_i \overset{\#}{\hookrightarrow} s_i$ (in $\mathcal{L}'$) there is a transition $T_{3-i} \overset{\#}{\hookrightarrow} s_{3-i}$ such that $s_1 \sim_n s_2$. Assuming that $s_1 \sim_n s_2$ iff $s_1 \sim'_{3n} s_2$ (Claim 1), we thus get that $T_1 \sim_n T_2$ iff $T_1 \sim'_{3n+1} T_2$ (Claim 3).
- If Claim 3 holds for n, then Claim 2 holds for n:

  Let $d_1 \sim_n d_2$, $i \in \{1, 2\}$, and $d_i \overset{\rho}{\hookrightarrow} T_i$ (hence $d_i(T_i) \geq \rho$). Put $T_{3-i} = \{ s' \in \mathit{support}(d_{3-i}) \mid s' \sim_n s$ for some $s \in T_i \}$. For each $\sim_n$-class E we have $d_1(E) = d_2(E)$, and thus $d_i(T_i \cap E) \leq d_{3-i}(T_{3-i} \cap E)$. Hence $d_i(T_i) \leq d_{3-i}(T_{3-i})$, and thus $d_{3-i} \overset{\rho}{\hookrightarrow} T_{3-i}$; moreover, in T_1, T_2 the same $\sim_n$-classes are represented, i.e., $T_1 \sim_n T_2$, hence $T_1 \sim'_{3n+1} T_2$ (assuming Claim 3). Therefore $d_1 \sim'_{1+(3n+1)} d_2$, i.e., $d_1 \sim'_{3n+2} d_2$.

  Now let $d_1 \not\sim_n d_2$; there is thus a $\sim_n$-class E such that $d_i(E) > d_{3-i}(E)$ for some $i \in \{1, 2\}$. For the transition $d_i \overset{\rho}{\hookrightarrow} T_i$ where $T_i = \mathit{support}(d_i) \cap E$ and $\rho = d_i(T_i)$ and any transition $d_{3-i} \overset{\rho}{\hookrightarrow} T_{3-i}$ we have $T_1 \not\sim_n T_2$; indeed, from $d_{3-i}(T_{3-i}) \geq \rho$ we deduce $d_{3-i}(T_{3-i}) \geq \rho = d_i(T_i) = d_i(E) > d_{3-i}(E)$, and the fact that $T_{3-i} \subseteq \mathit{support}(d_{3-i})$ then entails the existence of some $s \in T_{3-i} \smallsetminus E$. Since $T_1 \not\sim_n T_2$ entails $T_1 \not\sim'_{3n+1} T_2$ (assuming Claim 3), we get $d_1 \not\sim'_{1+(3n+1)} d_2$, i.e., $d_1 \not\sim'_{3n+2} d_2$.
- If Claim 2 holds for n, then Claim 1 holds for $n{+}1$:

  By definition (of relations $\sim_n$ in a pLTS), $s_1 \sim_{n+1} s_2$ iff for each $i \in \{1, 2\}$ and each transition $s_i \xrightarrow{a} d_i$ (in $\mathcal{L}$) there is a transition $s_{3-i} \xrightarrow{a} d_{3-i}$ such that $d_1 \sim_n d_2$. By definition of $\mathcal{L}'$ the only transitions from a state $s \in S$ in $\mathcal{L}'$ are the transitions $s \overset{a}{\hookrightarrow} d$ where $s \xrightarrow{a} d$ in $\mathcal{L}$. With the assumption that $d_1 \sim_n d_2$ iff $d_1 \sim'_{3n+2} d_2$ (Claim 2) we thus easily verify that $s_1 \sim_{n+1} s_2$ iff $s_1 \sim'_{1+(3n+2)} s_2$, i.e., $s_1 \sim_{n+1} s_2$ iff $s_1 \sim'_{3(n+1)} s_2$. □

Remark. We have confined ourselves to pLTSs $\mathcal{L} = (S, \Sigma, \rightarrow)$ that are image-finite and where Σ is finite and S is at most countable. This framework is sufficient for our intended applications, but the above transformation of a pLTS $\mathcal{L}$ to the LTS $\mathcal{L}'$ can be applied to general pLTSs as well (where also equivalences $\sim_\lambda$ for infinite ordinals λ are considered).

4. Upper Bounds for Subclasses of pPDA

The transformation of a pLTS $\mathcal{L}$ to an LTS $\mathcal{L}'$ described in Section 3 is now applied to derive decidability and complexity results for bisimilarity for pPDA (and subclasses thereof) from known results for standard (i.e., non-probabilistic) versions.

We use the fact that if $\mathcal{L}$ is generated by a probabilistic model M with a finite control unit or a stack (which is, in particular, the case of pPDA, pBPA, pOCA, pvPDA), then there are straightforward transformations of M to non-probabilistic versions M' that represent $\mathcal{L}'$. An explicit presentation of M' might be exponentially larger than M, so using complexity results for standard models as a "black box" to derive complexity results for probabilistic models incurs an exponential complexity blow-up. This complexity increase is not significant for general pPDA, since bisimilarity for PDA is known to be non-elementary [BGKM13] (it is even Ackermann-hard [Jan14b] for the model studied in [Sén05]). On the other hand, in the cases of standard OCA, vPDA, BPA, the known upper bounds are PSPACE, EXPTIME, and 2-EXPTIME, respectively. (For OCA and vPDA the upper bounds match the known lower bounds, the problem for BPA is only known to be EXPTIME-hard.)

Since M' arises from M by a specific enhancement causing only a "local" exponential increase that can be left implicit (i.e., M can be viewed as representing M' without an explicit construction of M'), by studying the algorithms for standard cases we might be able to verify that the exponential increase can be avoided. We will show that this is indeed the case for pvPDA, pBPA, and pOCA: here we will argue that the exponential increase only plays a role in the changed protocol (that transforms a current pair of states to a new one), which can be still performed in polynomial time with bounded alternation and does not affect the mentioned PSPACE, EXPTIME, 2-EXPTIME upper bounds. In the general case of pPDA we use the "black-box" reduction (in Section 4.1), since a similar argument that the complexity bound does not increase would require recalling the involved algorithms in the standard case [Sén05, Jan14a] and, as we have mentioned above, the complexity is already non-elementary in the standard case.

4.1. **Bisimilarity of pPDA is Decidable.** As announced, here we show a "black-box" use of the above transformation in the general case of pPDA. We have introduced PDA as pPDA with only Dirac distributions. In fact, PDA are the standard nondeterministic pushdown automata (with no ε-transitions). For PDA, bisimilarity is known to be decidable [Sén05] (see also [Jan14a]), with no explicit complexity upper bound. To show the decidability of bisimilarity for pPDA, it suffices to show how to realize a transformation of a pPDA Δ, representing a pLTS $\mathcal{L}$, to a PDA Δ' representing the LTS $\mathcal{L}'$ as defined in Section 3.

This task is straightforward, since we can naturally represent the relevant distributions and the subsets of their supports by using either the control state unit or (the top of) the stack. Because of the mentioned subsets of supports, we can thus increase the size of the control unit or of the stack alphabet exponentially; the size of the action alphabet increases similarly (due to actions $\rho \in \mathbb{Q}$). We now describe the "stack option" in detail.

For a pPDA $\Delta = (Q, \Gamma, \Sigma, \hookrightarrow)$ we define the PDA $\Delta' = (Q, \Gamma', \Sigma', \multimap\!\!\to)$ as follows:

- The stack alphabet Γ' arises from Γ by adding the following fresh symbols: for every distribution d such that Δ contains a rule $pX \stackrel{a}{\hookrightarrow} d$ we add a symbol $\langle d \rangle$; for every nonempty set T such that $T \subseteq \mathit{support}(d)$ for some of the above distributions d we add a symbol $\langle T \rangle$.
- $\Sigma' = \Sigma \cup W \cup \{\#\}$ where the parts of the union are pairwise disjoint and $W = \{\, \rho \in \mathbb{Q} \mid \rho = d(T)$ for some rule $pX \stackrel{a}{\hookrightarrow} d$ and $\emptyset \neq T \subseteq \mathit{support}(d) \,\}$.
- The set $\multimap\!\!\to$ of rules of Δ' is defined as follows. We choose an arbitrary state $q_0 \in Q$. Every rule $qX \stackrel{a}{\hookrightarrow} d$ of Δ gives rise to the following rules of Δ':

– $qX \overset{a}{\circ\!\!\to} q_0\langle d\rangle$;
– $q_0\langle d\rangle \overset{\rho}{\circ\!\!\to} q_0\langle T\rangle$ for all $\rho \in W$ and $T \subseteq \mathit{support}(d)$ where $d(T) \geq \rho$;
– $q_0\langle T\rangle \overset{\#}{\circ\!\!\to} p\alpha$ for each above defined symbol $\langle T\rangle$ and $p\alpha \in T$.

Example 4.1. Consider the pPDA $\Delta = (\{p,q\},\{X,Y\},\{a,b\},\hookrightarrow)$ with the following rules:

$$pX \overset{a}{\hookrightarrow} \tfrac{1}{3}q + \tfrac{2}{3}pYX, \qquad qY \overset{a}{\hookrightarrow} \tfrac{1}{3}p + \tfrac{2}{3}pX, \qquad pY \overset{b}{\hookrightarrow} qY, \qquad qY \overset{a}{\hookrightarrow} pXY.$$

The construction above yields the PDA $\Delta' = (\{p,q\},\Gamma',\Sigma',\circ\!\!\to)$ where

$$\begin{aligned}\Gamma' &= \{X,Y\} \cup \{\langle \tfrac{1}{3}q + \tfrac{2}{3}pYX\rangle, \langle \tfrac{1}{3}p + \tfrac{2}{3}pX\rangle, \langle 1qY\rangle, \langle 1pXY\rangle\} \cup \\ &\quad \cup \{\langle\{q,pYX\}\rangle, \langle\{q\}\rangle, \langle\{pYX\}\rangle, \langle\{p,pX\}\rangle, \langle\{p\}\rangle, \langle\{pX\}\rangle, \langle\{qY\}\rangle, \langle\{pXY\}\rangle\},\\ \Sigma' &= \{a,b,\tfrac{1}{3},\tfrac{2}{3},1,\#\},\end{aligned}$$

and, choosing $q_0 = p$, the rules in $\circ\!\!\to$ are given in Figure 4.

$$pX \overset{a}{\circ\!\!\to} p\langle \tfrac{1}{3}q + \tfrac{2}{3}pYX\rangle \qquad qY \overset{a}{\circ\!\!\to} p\langle \tfrac{1}{3}p + \tfrac{2}{3}pX\rangle \qquad pY \overset{b}{\circ\!\!\to} p\langle 1qY\rangle \qquad qY \overset{a}{\circ\!\!\to} p\langle 1pXY\rangle$$

$$\begin{array}{rcl}
p\langle \tfrac{1}{3}q + \tfrac{2}{3}pYX\rangle \overset{1}{\circ\!\!\to} p\langle\{q,pYX\}\rangle & \qquad & p\langle \tfrac{1}{3}p + \tfrac{2}{3}pX\rangle \overset{1}{\circ\!\!\to} p\langle\{p,pX\}\rangle \\
p\langle \tfrac{1}{3}q + \tfrac{2}{3}pYX\rangle \overset{\frac{2}{3}}{\circ\!\!\to} p\langle\{q,pYX\}\rangle & & p\langle \tfrac{1}{3}p + \tfrac{2}{3}pX\rangle \overset{\frac{2}{3}}{\circ\!\!\to} p\langle\{p,pX\}\rangle \\
p\langle \tfrac{1}{3}q + \tfrac{2}{3}pYX\rangle \overset{\frac{1}{3}}{\circ\!\!\to} p\langle\{q,pYX\}\rangle & & p\langle \tfrac{1}{3}p + \tfrac{2}{3}pX\rangle \overset{\frac{1}{3}}{\circ\!\!\to} p\langle\{p,pX\}\rangle \\
p\langle \tfrac{1}{3}q + \tfrac{2}{3}pYX\rangle \overset{\frac{1}{3}}{\circ\!\!\to} p\langle\{q\}\rangle & & p\langle \tfrac{1}{3}p + \tfrac{2}{3}pX\rangle \overset{\frac{1}{3}}{\circ\!\!\to} p\langle\{p\}\rangle \\
p\langle \tfrac{1}{3}q + \tfrac{2}{3}pYX\rangle \overset{\frac{2}{3}}{\circ\!\!\to} p\langle\{pYX\}\rangle & & p\langle \tfrac{1}{3}p + \tfrac{2}{3}pX\rangle \overset{\frac{2}{3}}{\circ\!\!\to} p\langle\{pX\}\rangle \\
p\langle \tfrac{1}{3}q + \tfrac{2}{3}pYX\rangle \overset{\frac{1}{3}}{\circ\!\!\to} p\langle\{pYX\}\rangle & & p\langle \tfrac{1}{3}p + \tfrac{2}{3}pX\rangle \overset{\frac{1}{3}}{\circ\!\!\to} p\langle\{pX\}\rangle \\
p\langle 1qY\rangle \overset{1}{\circ\!\!\to} p\langle\{qY\}\rangle & & p\langle 1pXY\rangle \overset{1}{\circ\!\!\to} p\langle\{pXY\}\rangle \\
p\langle 1qY\rangle \overset{\frac{2}{3}}{\circ\!\!\to} p\langle\{qY\}\rangle & & p\langle 1pXY\rangle \overset{\frac{2}{3}}{\circ\!\!\to} p\langle\{pXY\}\rangle \\
p\langle 1qY\rangle \overset{\frac{1}{3}}{\circ\!\!\to} p\langle\{qY\}\rangle & & p\langle 1pXY\rangle \overset{\frac{1}{3}}{\circ\!\!\to} p\langle\{pXY\}\rangle \\
p\langle\{q,pYX\}\rangle \overset{\#}{\circ\!\!\to} q & & p\langle\{p,pX\}\rangle \overset{\#}{\circ\!\!\to} p \\
p\langle\{q,pYX\}\rangle \overset{\#}{\circ\!\!\to} pYX & & p\langle\{p,pX\}\rangle \overset{\#}{\circ\!\!\to} pX \\
p\langle\{q\}\rangle \overset{\#}{\circ\!\!\to} q & & p\langle\{p\}\rangle \overset{\#}{\circ\!\!\to} p \\
p\langle\{pYX\}\rangle \overset{\#}{\circ\!\!\to} pYX & & p\langle\{pX\}\rangle \overset{\#}{\circ\!\!\to} pX \\
p\langle\{qY\}\rangle \overset{\#}{\circ\!\!\to} qY & & p\langle\{pXY\}\rangle \overset{\#}{\circ\!\!\to} pXY
\end{array}$$

Figure 4: Set $\circ\!\!\to$ of rules of the PDA Δ' from Example 4.1

Remark. We note that the choice of q_0 plays no role. We could introduce a fresh "don't care" state, but the choice of q_0 from Q makes clear that the original state set Q need not be extended. Hence if Δ is a pBPA then Δ' is a BPA.

Let $\mathcal{L} = \mathcal{L}_\Delta$ be the pLTS generated by a pPDA Δ. The (standard) LTS $\mathcal{L}_{\Delta'}$ defined by Δ' is bigger than $\mathcal{L}'$ as defined in Section 3, since the new stack symbols $\langle d \rangle$ and $\langle T \rangle$ can occur anywhere in the stack in configurations of Δ', not only at the top as intended. But if we restrict $\mathcal{L}_{\Delta'}$ to the states of $\mathcal{L}_\Delta$ (i.e., to the configurations of Δ), and close this set under reachability (in $\mathcal{L}_{\Delta'}$), then we obviously get an LTS that is isomorphic with $\mathcal{L}'$. Hence by Lemma 3.3 we deduce the following theorem:

Theorem 4.2. *For any pPDA Δ there is a PDA Δ' constructible in exponential time such that for any configurations $q_1\gamma_1, q_2\gamma_2$ of Δ we have $q_1\gamma_1 \sim q_2\gamma_2$ in $\mathcal{L}_\Delta$ if and only if $q_1\gamma_1 \sim q_2\gamma_2$ in $\mathcal{L}_{\Delta'}$. Hence bisimilarity for pPDA is decidable.*

In Sections 4.2 and 4.3 we will refer to the above "stack-version" PDA Δ' constructed to a given pPDA Δ. In Section 4.4 we will use the following "control-state version" Δ'_{C}.

For a pPDA $\Delta = (Q, \Gamma, \Sigma, \hookrightarrow)$, the PDA $\Delta'_{\mathrm{C}} = (Q', \Gamma, \Sigma', \circ\!\!\to)$ arises analogously as Δ' but with the following modifications:

- The symbols $\langle d \rangle$, $\langle T \rangle$ are added to Q instead of Γ.
- Every rule $qX \overset{a}{\hookrightarrow} d$ of Δ give rise to the following rules of Δ'_{C}:

  $qX \overset{a}{\circ\!\!\to} \langle d \rangle X$; $\langle d \rangle X \overset{\rho}{\circ\!\!\to} \langle T \rangle X$ (for ρ and T as in Δ'); $\langle T \rangle X \overset{\#}{\circ\!\!\to} p\alpha$ for $p\alpha \in T$.

We note that if $\mathcal{L} = \mathcal{L}_\Delta$ then $\mathcal{L}_{\Delta'_{\mathrm{C}}}$ gets isomorphic with the LTS $\mathcal{L}'$ defined in Section 3 when we identify its states $\langle d \rangle X\gamma$, $\langle d \rangle Y\gamma$ and also $\langle T \rangle X\gamma$, $\langle T \rangle Y\gamma$. In other words, in the configurations $\langle d \rangle \alpha$, $\langle T \rangle \alpha$ the first stack symbol plays no role. We also note that Δ'_{C} is an OCA when Δ is a pOCA.

Remark. We will show that known algorithms for vPDA, BPA, OCA in principle also work for pvPDA, pBPA, pOCA with no substantial complexity increase, since they can be viewed as working on "big" PDA Δ' or Δ'_{C} though these big PDA will be only implicitly presented by "small" pPDA Δ. We could try to define a more abstract notion of "concise PDA" that represent bigger standard PDA so that the algorithms for standard cases would also work for concise PDA without a substantial complexity increase, and pvPDA, pBPA, pOCA would be just special cases of such concise PDA. But we leave such an abstraction as a mere possibility, since it would add further technicalities.

4.2. **Bisimilarity of pvPDA is in EXPTIME.** It is shown in [Srb09, Theorem 3.3] that the bisimilarity problem for (standard) vPDA is EXPTIME-complete. We will show that the same holds for pvPDA. In this section we show the upper bound:

Theorem 4.3. *The bisimilarity problem for pvPDA is in EXPTIME.*

In [Srb09] the upper bound is proved by a reduction to the model-checking problem for (general) PDA and the modal μ-calculus; the latter problem is in EXPTIME by [Wal01]. This reduction does not apply in the probabilistic case, and if we use the reduction from Section 3 explicitly (as in Section 4.1) and apply the result of [Wal01] to the resulting exponentially bigger instance, then we only derive a double-exponential upper bound. In fact, if for a given pvPDA Δ we construct the PDA Δ' as in Section 4.1, then Δ' might be formally not a vPDA (since the action # can have different effects on the stack height), but it is straightforward to adjust the construction so that Δ' becomes a vPDA.

Nevertheless, we give a self-contained proof that the bisimilarity problem for pvPDA is in EXPTIME; since vPDA is a special case, we thus also provide a new proof of the EXPTIME upper bound in the standard case.

Proof. We consider a pvPDA $\Delta = (Q, \Gamma, \Sigma, \hookrightarrow)$ with the respective partition $\Sigma = \Sigma_r \cup \Sigma_{int} \cup \Sigma_c$; it generates the respective pLTS $\mathcal{L}_\Delta$. By Δ' we refer to the (standard) PDA arising as in Section 4.1, but we do not assume constructing it explicitly. We recall that $p\alpha \sim q\beta$ in $\mathcal{L}_\Delta$ iff $p\alpha \sim q\beta$ in $\mathcal{L}_{\Delta'}$.

For each $a \in \Sigma$ we now define a relation $\overset{a}{\longmapsto}$ (not necessarily a function) between pairs and sets of pairs: For $a \in \Sigma$, $p, q \in Q$, $X, Y \in \Gamma$, and $Out \subseteq (Q\Gamma^{\leq 2}) \times (Q\Gamma^{\leq 2})$ we put

$$(pX, qY) \overset{a}{\longmapsto} Out$$

if the following conditions hold:

(1) In the protocol running from (pX, qY), i.e., in the respective three-round bisimulation game, Attacker can use an a-transition so that it is guaranteed that Defender either loses (having no response with action a) or the outcome is a pair from Out.
(2) If $a \in \Sigma_c$, then $Out \subseteq Q\Gamma\Gamma \times Q\Gamma\Gamma$; if $a \in \Sigma_{int}$, then $Out \subseteq Q\Gamma \times Q\Gamma$; if $a \in \Sigma_r$, then $Out \subseteq Q \times Q$.

We can easily verify that all relations $\overset{a}{\longmapsto}$ can be constructed in exponential time; indeed, to verify that $(pX, qY) \overset{a}{\longmapsto} Out$ for concrete p, q, X, Y, a, Out, it suffices to use a polynomial-time algorithm with bounded alternation (that simulates three rounds of the bisimulation game in $\mathcal{L}_{\Delta'}$ starting in (pX, qY)).

Now by $\rightsquigarrow$ we denote the least relation between the set $Q\Gamma \times Q\Gamma$ and the set $2^{Q\times Q}$ (of subsets of $(Q \times Q)$) satisfying the following (inductive) conditions:

(1) if $(pX, qY) \overset{a}{\longmapsto} Out$ for $a \in \Sigma_r$ (hence $Out \subseteq Q \times Q$), then $(pX, qY) \rightsquigarrow Out$;
(2) if $(pX, qY) \overset{a}{\longmapsto} Out_1$ for $a \in \Sigma_{int}$, $Out_2 \subseteq Q \times Q$, and $(p'X', q'Y') \rightsquigarrow Out_2$ for each $(p'X', q'Y') \in Out_1$, then $(pX, qY) \rightsquigarrow Out_2$;
(3) if $(pX, qY) \overset{a}{\longmapsto} Out_1$ for $a \in \Sigma_c$, $Out_2 \subseteq Q \times Q$, and for each $(p'X_1X_2, q'Y_1Y_2) \in Out_1$ there is $Out' \subseteq Q \times Q$ such that $(p'X_1, q'Y_1) \rightsquigarrow Out'$ and $(p''X_2, q''Y_2) \rightsquigarrow Out_2$ for each $(p'', q'') \in Out'$, then $(pX, qY) \rightsquigarrow Out_2$.

It is obvious that $(pX, qY) \rightsquigarrow Out$ implies that Attacker has a strategy guaranteeing that in each play starting in (pX, qY) he either wins or a pair $(p'\varepsilon, q'\varepsilon)$ with $(p', q') \in Out$ is reached (when he adheres to the strategy). The opposite implication also holds, since if Attacker has such a strategy, then he can guarantee reaching his goal in n steps, for some $n \in \mathbb{N}$. (This follows from image-finiteness, which entails that Defender always has a finite, even bounded, number of possible responses.)

To construct the set $\rightsquigarrow \subseteq (Q\Gamma \times Q\Gamma) \times 2^{Q\times Q}$, we can use the standard least-fixed-point computation: we put $\rightsquigarrow_0 = \emptyset$, and from $\rightsquigarrow_i$ we get $\rightsquigarrow_{i+1} \supseteq \rightsquigarrow_i$ by applying the above "deduction rules" 1.–3. (replacing $\rightsquigarrow$ with $\rightsquigarrow_i$ in the antecedents and with $\rightsquigarrow_{i+1}$ in the consequents). This yields that $\rightsquigarrow$ can be constructed in exponential time w.r.t. the size of the given pvPDA Δ.

For an exponential algorithm deciding if $p\alpha \sim q\beta$ for given configurations $p\alpha$, $q\beta$, we use an inductive construction of the sets

$$\mathcal{A}(\alpha, \beta) = \{\, (p, q) \mid p\alpha \not\sim q\beta \,\}.$$

The construction is based on the following simple observations:

- $\mathcal{A}(\varepsilon, \varepsilon) = \emptyset$;
- $\mathcal{A}(\varepsilon, Y\beta') = \{ (p, q) \mid qY \text{ enables some action } a \in \Sigma \}$;
- $\mathcal{A}(X\alpha', Y\beta') = \{ (p, q) \mid (pX, qY) \rightsquigarrow \mathcal{A}(\alpha', \beta') \}$.

For deciding if $pX_1X_2 \cdots X_m \sim qY_1Y_2 \cdots Y_n$, where we assume $m \leq n$ w.l.o.g., we can stepwise construct the sets $\mathcal{A}_m = \mathcal{A}(\varepsilon, Y_{m+1}Y_{m+2} \cdots Y_n)$, $\mathcal{A}_{i-1} = \{ (p', q') \mid (p'X_i, q'Y_i) \rightsquigarrow \mathcal{A}_i \}$ for $i = m, m-1, \ldots, 1$, and finally give the answer YES if $(p, q) \notin \mathcal{A}_0$, and NO if $(p, q) \in \mathcal{A}_0$. □

Example 4.4. Consider the pvPDA $\Delta = (\{p, p', q, q'\}, \{X\}, \{c, r\}, \hookrightarrow)$ with the following rules:

$$pX \overset{c}{\hookrightarrow} \tfrac{1}{2}pXX + \tfrac{1}{2}p'XX, \qquad p'X \overset{c}{\hookrightarrow} p'XX, \qquad pX \overset{r}{\hookrightarrow} q, \qquad p'X \overset{r}{\hookrightarrow} q', \qquad qX \overset{r}{\hookrightarrow} q.$$

We have

$$(pX, p'X) \overset{c}{\longmapsto} \{(pXX, p'XX)\}, \qquad (pX, p'X) \overset{r}{\longmapsto} \{(q, q')\}, \qquad (qX, q'X) \overset{r}{\longmapsto} \emptyset,$$

and this is not an exhaustive list. The first deduction rule yields $(pX, p'X) \rightsquigarrow \{(q, q')\}$ and $(qX, q'X) \rightsquigarrow \emptyset$. By the third deduction rule we thus have $(pX, p'X) \rightsquigarrow \emptyset$. Hence $(p, p') \in \mathcal{A}(X, X)$, and so $pX \not\sim p'X$.

4.3. Bisimilarity of pBPA is in 2-EXPTIME.

Bisimilarity of (standard) BPA is known to be in 2-EXPTIME (as claimed in [BCMS01] and explicitly proved in [Jan13]). Here we show that the same upper bound applies to pBPA as well.

We first recall that if Δ is a pBPA then the pPDA Δ' defined in Section 4.1 is a BPA (since Δ and Δ' have the same singleton state sets). Hence Theorem 4.2, with the 2-EXPTIME result for BPA, immediately yields a triple-exponential upper bound for pBPA. To argue that this exponential increase is not necessary, we recall the proof from [Jan13] and show that mild modifications yield a double-exponential algorithm also for pBPA. We assume the reader has access to [Jan13], and we thus only recall the parts of the proof relevant to the generalization rather than repeating the whole argument in all detail. Nevertheless, we also try to convey the intuitive ideas from [Jan13] to facilitate understanding.

Theorem 4.5. *The bisimilarity problem for pBPA is in 2-EXPTIME.*

Proof. We consider a pBPA $\Delta = (\Gamma, \Sigma, \hookrightarrow)$, omitting the singleton state set Q; it generates the respective pLTS $\mathcal{L}_\Delta$. By Δ' we refer to the (standard) BPA arising as in Section 4.1, but we do not assume constructing it explicitly. We recall that

$$\alpha \sim_n \beta \text{ in } \mathcal{L}_\Delta \text{ iff } \alpha \sim_{3n} \beta \text{ in } \mathcal{L}_{\Delta'} \tag{4.1}$$

(using Lemma 3.3). One important fact is that the relations $\sim_n$ and $\sim$ are congruences w.r.t. concatenation, i.e.: $\alpha \sim_n \alpha'$ and $\beta \sim_n \beta'$ imply $\alpha\beta \sim_n \alpha'\beta'$. This holds in the LTS $\mathcal{L}_{\Delta'}$ by Proposition 3.1 in [Jan13], and thus also in the pLTS $\mathcal{L}_\Delta$ (due to (4.1)), when assuming that each $X \in \Gamma$ enables at least one action (cf. the Remark after Proposition 3.1 in [Jan13]).

To match the approach in [Jan13], we assume that the states in $\mathcal{L}_\Delta$ are not only finite strings from Γ^* but also infinite regular (i.e., ultimately periodic) strings from Γ^ω.

In Section 3.2 of [Jan13] the main algorithm is described in the form of a Prover-Refuter game. To give an intuition, we consider a pair of the form $(A\alpha, B\beta)$ for which Prover claims $A\alpha \sim B\beta$ and Refuter claims $A\alpha \not\sim B\beta$; we have $A, B \in \Gamma$ and $\alpha, \beta \in \Gamma^* \cup \Gamma^\omega$. (The notation in [Jan13] uses $\mathcal{G} = (\mathcal{N}, \mathcal{A}, \mathcal{R})$ for BPA.) By the (Prover-Refuter) game protocol, when Δ is a BPA, Prover has the option to decide that one round of the standard bisimulation game, played by Attacker and Defender, will be mimicked: in this case Refuter performs an Attacker's move, from the pair-component $A\alpha$ or $B\beta$, and Prover responds from the other pair-component by a move with the same action. We thus get two moves $A\alpha \xrightarrow{a} \gamma_1\alpha$ and $B\beta \xrightarrow{a} \gamma_2\beta$ where Prover claims $\gamma_1\alpha \sim \gamma_2\beta$ and Refuter claims that $\textsc{EqLv}(\gamma_1\alpha, \gamma_2\beta) < \textsc{EqLv}(A\alpha, B\beta)$, where the *equivalence-level* is defined as $\textsc{EqLv}(\delta_1, \delta_2) = \max\{n \mid \delta_1 \sim_n \delta_2\}$. The play then continues with the pair $(\gamma_1\alpha, \gamma_2\beta)$.

The described option of one round of the bisimulation game corresponds to Point (3c) in Section 3.2 of [Jan13]. Now, when Δ is a pBPA, the respective option is that Prover and Refuter mimic one round of the "probabilistic game", i.e., of three rounds of the bisimulation game in $\mathcal{L}_{\Delta'}$, which also results in a new pair $(\gamma_1\alpha, \gamma_2\beta)$ of states from $\mathcal{L}_\Delta$, as above. Such three rounds can be performed in polynomial time (w.r.t. the size of the pBPA Δ) with bounded alternation. Since we discuss an alternating-expspace complexity bound (recalling that AEXPSPACE=2-EXPTIME), the described change of (3c) causes no problem.

Another option that Prover has for the pair $(A\alpha, B\beta)$ is to provide several relevant smaller pairs from which Refuter will choose one to continue. Concretely, Prover can decide to provide two or three smaller pairs in the forms captured by the following three possibilities:

(1) $(\alpha, \gamma\beta)$, $(A\gamma, B)$;
(2) $(\alpha, \gamma\beta)$, (β, δ^ω), $(A\gamma\delta^\omega, B\delta^\omega)$;
(3) $(\alpha, (\gamma\delta)^\omega)$, $(\beta, (\delta\gamma)^\omega)$, $(A(\gamma\delta)^\omega, B(\delta\gamma)^\omega)$.

(As expected, $\delta^\omega = \delta\delta\delta\cdots$.) In each of the above cases, $(A\alpha, B\beta)$ belongs to the least congruence containing the respective (two or three) pairs; this entails that the least equivalence-level of the respective pairs cannot be bigger than $\textsc{EqLv}(A\alpha, B\beta)$. The size of finite strings from Γ^* and regular strings from Γ^ω is based on the notion of norms. By the *norm* $\|A\|$, for $A \in \Gamma$, we mean the length of a shortest u such that $A \xrightarrow{u} \varepsilon$; if there is no such u, then A is *unnormed* and we put $\|A\| = \omega$. We can imagine that we refer to the BPA Δ' when computing $\|A\|$. But since each (completed) path in $\mathcal{L}_{\Delta'}$ that starts from a state in $\mathcal{L}_\Delta$ is composed of segments $X\alpha \xrightarrow{a} \langle d\rangle\alpha \xrightarrow{\rho} \langle T\rangle\alpha \xrightarrow{\#} \gamma\alpha$ (where $X \xhookrightarrow{a} d$ is a rule of Δ and $d(\gamma) > 0$), the finite norms $\|A\|$, $A \in \Gamma$, are at most exponential, and computable in polynomial time, *w.r.t. the size of the pBPA* Δ (as easily follows by mimicking the proof of Proposition 3.6. in [Jan13]). We also put $\|\varepsilon\| = 0$ and $\|A\alpha\| = \|A\| + \|\alpha\|$, when $\|A\| < \omega$. Since $U\alpha \sim U$ when U is unnormed, we only consider the strings of the form α, αU, $\alpha(\beta)^\omega$ such that α and β are normed strings from Γ^* (i.e., $\|\alpha\|$ and $\|\beta\|$ are finite) and $U \in \Gamma$ is unnormed. In these cases we put $\textsc{Size}(\alpha) = \textsc{Size}(\alpha U) = \|\alpha\|$ and $\alpha(\beta)^\omega = \textsc{Size}(\alpha) + \textsc{Size}(\beta)$ if α, β constitute the canonical presentation of the regular infinite string $\alpha(\beta)^\omega$ (as defined in a standard way in [Jan13]). We also put $\textsc{Size}(\delta_1, \delta_2) = \max\{\textsc{Size}(\delta_1), \textsc{Size}(\delta_2)\}$.

Lemma 3.14 in [Jan13] shows that if $\textsc{Size}(A\alpha, B\beta)$ is bigger than an exponential constant and $A\alpha \sim B\beta$, then there is a decomposition of one of the above types 1,2,3 consisting of two or three pairs of the type (δ_1, δ_2) where $\delta_1 \sim \delta_2$ and $\textsc{Size}(\delta_1, \delta_2) < \textsc{Size}(A\alpha, B\beta)$. We can apply the mentioned lemma to the BPA Δ'; since the respective exponential constant is related to maximal finite norms $\|A\|$, it is exponential in the size of the pBPA Δ.

Hence exponential space is sufficient to keep the current game configuration so that Prover has a winning strategy for (δ_1, δ_2) iff $\delta_1 \sim \delta_2$. (Prover wins a possibly infinite play if she is always able to respond when Refuter is mimicking Attacker in the bisimulation game and always keeps the game configuration in the determined exponential bounds, by using a decomposition when a large pair arises.)

In the above conclusion, a subtle point is left implicit. We apply Lemma 3.14 from [Jan13] to a pair $(A\alpha, B\beta)$ of states in $\mathcal{L}_\Delta$ as to a pair of states in $\mathcal{L}_{\Delta'}$, and assume that we can choose the respective decompositions that consist of pairs of states in $\mathcal{L}_\Delta$. This cannot be deduced just from the statement of the lemma, but it suffices to perform a routine check of the proofs of Lemma 3.14 and the related Lemma 3.11 from [Jan13] to verify that such decompositions indeed exist. The only point in the proofs that might be not so straightforward is in (1) of the proof of Lemma 3.11 in [Jan13]. There we read

> We fix a rule $A_j \xrightarrow{a} \sigma_j$ such that for any rule $A_{3-j} \xrightarrow{a} \sigma_{3-j}$ we get $\textsc{EqLv}(A_1\gamma, A_2) > \textsc{EqLv}(\sigma_1\gamma, \sigma_2) \ldots$ Now we fix a rule $A_{3-j} \xrightarrow{a} \sigma_{3-j}$ such that $\sigma_1\gamma\mu_i\beta \sim \sigma_2\mu_i\beta$.

Instead we now say:

> We fix a strategy of Attacker in the three-round bisimulation game from (A_1, A_2) such that the outcome (σ_1, σ_2) (for whatever strategy of Defender) will satisfy $\textsc{EqLv}(A_1\gamma, A_2) > \textsc{EqLv}(\sigma_1\gamma, \sigma_2) \ldots$ Now we fix a strategy of Defender, in the three-round bisimulation game from (A_1, A_2), such that the outcome (σ_1, σ_2) will satisfy $\sigma_1\gamma\mu_i\beta \sim \sigma_2\mu_i\beta$. □

4.4. **Bisimilarity of pOCA is in PSPACE.** The bisimilarity problem for (standard) one-counter automata is known to be PSPACE-complete. We now argue that the decision algorithm described in [BGJ14] (originating in [BGJ10]) also shows, in fact, that the problem for pOCA is in PSPACE as well. We thus show:

Theorem 4.6. *The bisimilarity problem for pOCA is in PSPACE, even if we present the instance $\Delta = (Q, \{I, Z\}, \Sigma, \hookrightarrow)$, pI^mZ, qI^nZ (for which we ask if $pI^mZ \sim qI^nZ$) by a shorthand using m, n written in binary.*

Proof. Let us consider a pOCA $\Delta = (Q, \{I, Z\}, \Sigma, \hookrightarrow)$. Let $\Delta'_{\mathrm{C}} = (Q', \{I, Z\}, \Sigma', \circ\!\!\rightarrow)$ be the OCA defined in Section 4.1 (i.e., the "control-state version" of the respective PDA); hence we have

$$pI^mZ \sim_n qI^{m'}Z \text{ in } \mathcal{L}_\Delta \text{ iff } pI^mZ \sim_{3n} qI^{m'}Z \text{ in } \mathcal{L}_{\Delta'_{\mathrm{C}}}.$$

We now aim to apply the algorithm from [BGJ14] to Δ'_{C} but using only ("small") Δ as a presentation of ("big") Δ'_{C}. If the algorithm was applied to Δ'_{C} explicitly, it would construct a semilinear description of the mapping $\chi\colon \mathbb{N} \times \mathbb{N} \times (Q' \times Q') \to \{1, 0\}$ such that

$$\chi(m, n, (p, q)) = 1 \text{ iff } pI^mZ \sim qI^nZ.$$

But the set Q' can be exponentially larger than the size of Δ. The idea is that we simply let the algorithm compute just the restriction of χ to the domain $\mathbb{N} \times \mathbb{N} \times (Q \times Q)$; it will turn out that a description of this restricted mapping can be computed in polynomial space w.r.t. the size of Δ.

Generally, for any relation R on $Q \times (\{I\}^*Z)$, by the *(characteristic) colouring* χ_R we mean the mapping $\chi_R\colon \mathbb{N} \times \mathbb{N} \times (Q \times Q) \to \{1, 0\}$ where

$$\chi_R(m, n, (p, q)) = 1 \text{ iff } (pI^mZ, qI^nZ) \in R.$$

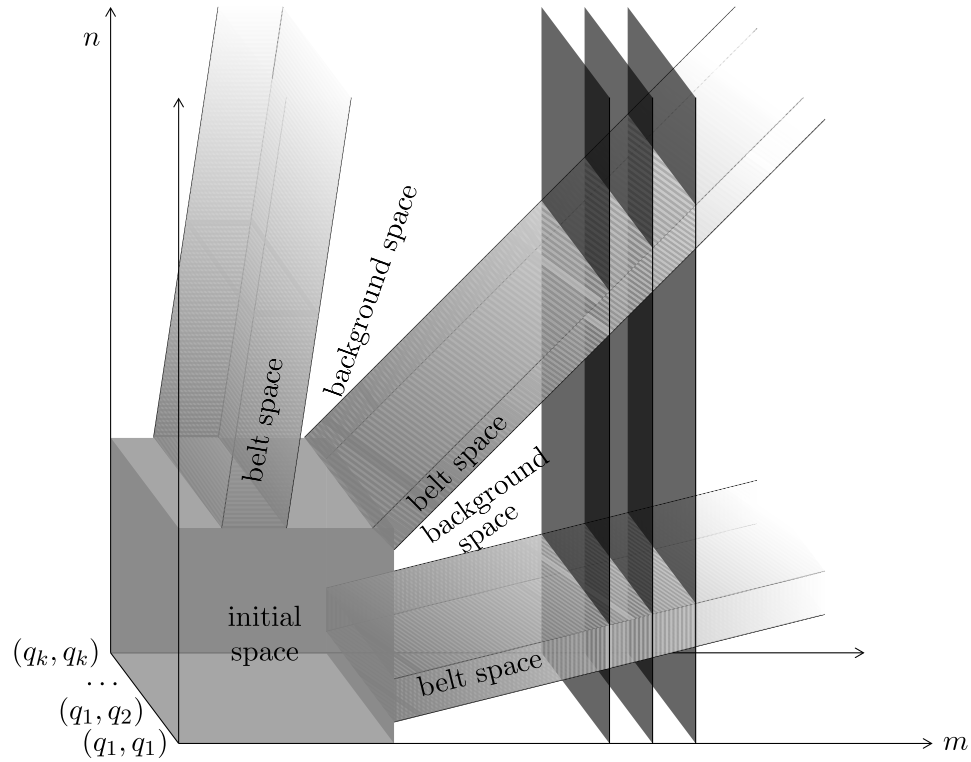


Figure 5: OCA: partition of a grid, and a moving vertical window of width 3

In Fig. 5 we can see a depiction of the domain $\mathbb{N}\times\mathbb{N}\times(Q\times Q)$, assuming $Q = \{q_1, q_2, \dots, q_k\}$. For a relation R on $Q\times(\{I\}^*Z)$, the mapping χ_R can be viewed as a "black-white colouring", making a point $(m, n, (q_i, q_j))$ black if $\chi_R(m, n, (q_i, q_j)) = 1$ and white if $\chi_R(m, n, (q_i, q_j)) = 0$. (See also the figures in [BGJ14].)

An important ingredient in [BGJ14] is the underlying finite LTS; here it underlies Δ'_{C}, and would be denoted by $\mathcal{F}_{\Delta'_{\mathrm{C}}}$ according to Section 3.2 in [BGJ14]. In our setting, we can describe $\mathcal{F}_{\Delta'_{\mathrm{C}}}$ as follows. We first consider the pLTS $\mathcal{L} = (Q, \Sigma, \to)$ with the following relation $\to$: each rule $pI \overset{a}{\hookrightarrow} d$ of Δ gives rise to the transition $p \overset{a}{\to} d'$ where $d'(q) = d(q\varepsilon) + d(qI) + d(qII)$ (for all $q \in Q$). Hence $\mathcal{L}$ behaves as Δ with the "always-positive" counter. In other words, we can consider extending the pLTS $\mathcal{L}_\Delta$ with the states qI^ω (for all $q \in Q$); then $\mathcal{L}$ is just the restriction to the set of these additional states (this set is closed under reachability). Now the mentioned $\mathcal{F}_{\Delta'_{\mathrm{C}}}$ is, in fact, the LTS $\mathcal{L}'$ corresponding to $\mathcal{L}$ as defined in Section 3.

The bisimulation equivalence on the finite pLTS $\mathcal{L}$ can be constructed in polynomial time by standard partition refinement technique [Bai96, BEM00]. In particular, if $|Q| = k$, then $\sim\ =\ \sim_{k-1}$ on $\mathcal{L}$. Though $\mathcal{L}'$ (i.e., $\mathcal{F}_{\Delta'_{\mathrm{C}}}$) is exponentially bigger than $\mathcal{L}$, due to its special form we have that

$$\sim\ =\ \sim_{3(k-1)} \text{ in } \mathcal{L}'.$$

Now the set INC from [BGJ14] of configurations "incompatible" with $\mathcal{L}'$ can be restricted to Q, and we thus put

$$\mathsf{INC} = \{\, pI^mZ \mid p \in Q, \forall q \in Q : pI^mZ \not\sim_{3k} q \text{ when } pI^mZ \text{ is from } \mathcal{L}_{\Delta'_{\mathrm{C}}} \text{ and } q \text{ from } \mathcal{L}' \,\}.$$

(We recall that comparing the states from different LTSs implicitly refers to the disjoint union of these LTSs.)

As in [BGJ14], we define $\mathsf{dist}(pI^mZ)$ as the length of a shortest word u such that $pI^mZ \xrightarrow{u} qI^nZ \in \mathsf{INC}$ in the LTS $\mathcal{L}_{\Delta'_{\mathrm{C}}}$; we put $\mathsf{dist}(pI^mZ) = \omega$ if there is no such word u.

The analysis in [BGJ14] applied to Δ'_{C}, $\mathcal{L}_{\Delta'_{\mathrm{C}}}$, $\mathcal{L}'$ gives us the following:

(1) If $m \geq 3k$, then $pI^mZ \sim_{3k} p$ where $p \in Q$, pI^mZ is viewed as a state of $\mathcal{L}_{\Delta'_{\mathrm{C}}}$, and p is viewed as a state of $\mathcal{L}'$. Hence $pI^mZ \in \mathsf{INC}$ implies $m < 3k$. (In fact, $3k$ can be replaced by k here, since for any segment $pI^mZ \xrightarrow{a} \langle d\rangle I^mZ \xrightarrow{\rho} \langle T\rangle I^mZ \xrightarrow{\#} qI^{m'}Z$ we have $m' \geq m{-}1$, but this is not important.)
(2) If $\mathsf{dist}(pI^mZ) = \omega$, then $pI^mZ \sim q$ for some q (even if $m < 3k$), and, moreover, $pI^mZ \sim q$ iff $pI^mZ \sim_{3k} q$.
(3) If $\mathsf{dist}(pI^mZ) \neq \mathsf{dist}(qI^nZ)$, then $pI^mZ \not\sim qI^nZ$.

We note that INC can be computed in polynomial time w.r.t. the size of Δ (though even polynomial space would suffice for us): by Point 1 we have $\mathsf{INC} \subseteq \{\, pI^mZ \mid p \in Q, m < 3k \,\}$, and deciding if $pI^mZ \sim_{3k} q$ can be easily done in polynomial time.

When computing $\mathsf{dist}(pI^mZ)$, we can consider only easily computable "macrosteps" $pI^mZ \to qI^{m'}Z$, with $p, q \in Q$ and $m' \in \{m{-}1, m, m{+}1\}$, where there are a,$\langle d\rangle$,ρ,$\langle T\rangle$ such that $pI^mZ \xrightarrow{a} \langle d\rangle I^mZ \xrightarrow{\rho} \langle T\rangle I^mZ \xrightarrow{\#} qI^{m'}Z$ in $\mathcal{L}_{\Delta'_{\mathrm{C}}}$.

Hence the analysis in [BGJ14] shows that the points $(m, n, (p,q))$ such that $\mathsf{dist}(pI^mZ) = \mathsf{dist}(qI^nZ) < \omega$ lie in "linear belts" (see Proposition 26 in [BGJ14]), i.e., in the "belt space" and the "initial space" depicted in Figure 5. Moreover, the belts have polynomial coefficients in the size of Δ: though formally we refer to Δ' with $|Q'|$ states, the coefficients are polynomial in the number $|Q|$ due to the above mentioned small number of macrosteps.

The polynomial space algorithm in [BGJ14] is "moving the vertical window of width 3" from the beginning to the right (as is also depicted in Figure 5 and in the figures in [BGJ14]). In principle it can guess the black points in the initial space and the belt space, while the colour of the points in the background space (where we either have $\mathsf{dist}(pI^mZ) \neq \mathsf{dist}(qI^nZ)$ or $\mathsf{dist}(pI^mZ) = \mathsf{dist}(qI^nZ) = \omega$) can be easily computed.

The algorithm now proceeds in the same way as Alg-Bisim in Section 4 of [BGJ14]; it guesses a black-white coloring that should represent a bisimulation but, as already mentioned, it guesses just the restriction to the domain $\mathbb{N} \times \mathbb{N} \times (Q \times Q)$. In Point (c)ii of Alg-Bisim it is checked if the guess in the intersection of the middle of the vertical window with the initial space and belt space is consistent w.r.t. the black-white colouring of its (closest) neighbourhood.

In our case, in such a black point $(m, n, (p,q))$ we simply run a polynomial-time algorithm with bounded alternation, mimicking three rounds of the bisimulation game from (pI^mZ, qI^nZ), to check if Defender can guarantee that the outcome is again a black point. $\square$

5. Lower Bounds

The upper bounds for pOCA and for pvPDA are tight: the bisimilarity problems already for standard versions are known to be PSPACE-hard for OCA (even for visibly OCA [Srb09])

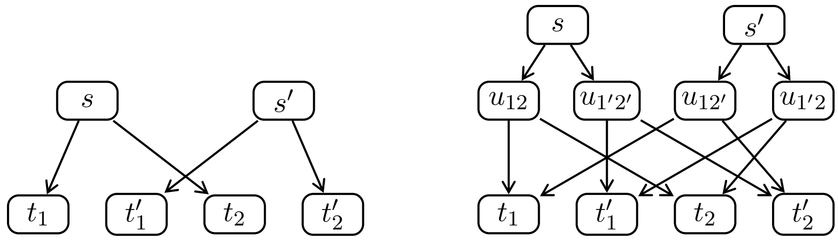


Figure 6: AND-gadget (left) and OR-gadget (right)

and EXPTIME-hard for vPDA (see [Srb09] where a relevant construction from [KM02] is used); hence in combination with Theorems 4.6 and 4.3 we obtain:

Corollary 5.1. *The bisimilarity problem for pOCA is PSPACE-complete, and the bisimilarity problem for pvPDA EXPTIME-complete.*

In Sections 5.1 and 5.2 we show that these lower bounds also apply to the fully probabilistic versions, even when the action alphabet is restricted to size 1 and 3, respectively. We define two gadgets, adapted from [CvBW12], that will be used for both results. The gadgets are small pLTSs that allow us to simulate AND and OR gates using probabilistic bisimilarity. We depict the gadgets in Figure 6, where we assume that all edges have probability $\frac{1}{2}$ and have the same label. The gadgets satisfy the following (trivially verifiable) propositions, in which we write $s \xrightarrow{a} t_1 \mid t_2$ as a shorthand for $s \xrightarrow{a} \frac{1}{2}t_1 + \frac{1}{2}t_2$.

Proposition 5.2. (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 5.3. (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*

$$s \xrightarrow{a} u_{12} \mid u_{1'2'} \qquad s' \xrightarrow{a} u_{12'} \mid u_{1'2}$$

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

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 5.4. *Bisimilarity is PSPACE-hard (even) for unary (i.e., with only one action) and fully probabilistic OCA.*

We remark that fully probabilistic PDA with only one action do not have any nondeterminism: they generate a countable-state Markov chain.

Proof. We use a reduction from the emptiness problem for alternating finite automata with a one-letter alphabet, known to be PSPACE-complete [Hol95, JS07]; our reduction resembles the reduction in [Srb09] 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* δ 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 now reduce this problem to our problem.

Assuming a 1L-AFA (Q, δ, q_0, F), we construct a pOCA $\Delta = (\overline{Q}, \{I, Z\}, \{a\}, \hookrightarrow)$ as follows. The state set $\overline{Q}$ contains $2|Q|{+}3$ 'basic' states; the set of basic states is $\{p_0, p_0', r\} \cup Q \cup Q'$ 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. We note that Δ has a singleton action alphabet, and it will be fully probabilistic. Below we describe a construction of $\hookrightarrow$, aiming to achieve $p_0 IZ \sim p_0' IZ$ if and only if $\{\, n \mid Acc(q_0, n) \,\} = \emptyset$; another property will be that

$$qI^n Z \sim q' I^n Z \text{ if and only if } \neg Acc(q, n). \tag{5.1}$$

For each $q \in F$ we create a rule $qZ \stackrel{a}{\hookrightarrow} rZ$, but qZ is dead (i.e., there is no rule $qZ \stackrel{a}{\hookrightarrow} \ldots$) if $q \notin F$; $q'Z$ is dead for any $q' \in Q'$. Both rI and rZ are dead as well. Hence (5.1) is satisfied for $n = 0$. Now we complete the set $\hookrightarrow$ of rules and show that (5.1) also holds for $n > 0$.

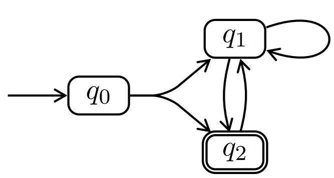
For $q \in Q$ with $\delta(q) = q_1 \vee q_2$ we implement an AND-gadget from Figure 6 (left) guaranteeing that $qI^{n+1}Z \sim q'I^{n+1}Z$ if and only if $q_1 I^n Z \sim q_1' I^n Z$ and $q_2 I^n Z \sim q_2' I^n Z$ (since $\neg Acc(q, n{+}1)$ if and only if $\neg Acc(q_1, n)$ and $\neg Acc(q_2, n)$):

We add rules $qI \to r_1 I \mid r_2 I$ (this is a shorthand for $qI \stackrel{a}{\hookrightarrow} \frac{1}{2} r_1 I + \frac{1}{2} r_2 I$) and $q'I \to r_1' I \mid r_2' I$, and also $r_1 I \to q_1 \mid s_1 I$, $r_2 I \to q_2 \mid s_2 I$, $r_1' I \to q_1' \mid s_1 I$, $r_2' I \to q_2' \mid s_2 I$, and finally $s_1 I \stackrel{a}{\hookrightarrow} \frac{1}{2} s_1 I + \frac{1}{2} r$, $s_2 I \stackrel{a}{\hookrightarrow} 0.4 s_2 I + 0.6 r$. 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 5.2.

For $q \in Q$ with $\delta(q) = q_1 \wedge q_2$ we (easily) implement an OR-gadget from Figure 6 (right) guaranteeing $qI^{n+1}Z \sim q'I^{n+1}Z$ if and only if $q_1 I^n Z \sim q_1' I^n Z$ or $q_2 I^n Z \sim q_2' I^n Z$ (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 rules $p_0 I \stackrel{a}{\hookrightarrow} \frac{1}{3} p_0 II + \frac{1}{3} q_0 \varepsilon + \frac{1}{3} rI$ and $p_0' I \stackrel{a}{\hookrightarrow} \frac{1}{3} p_0' II + \frac{1}{3} q_0' \varepsilon + \frac{1}{3} rI$. As $p_0 I$ and $p_0' I$ can transition to (the dead) rI, the rules added before guarantee that $p_0 I^{n+2} Z \not\sim q_0' I^n Z$ and $q_0 I^n Z \not\sim p_0' I^{n+2} Z$. □

Example 5.5. We illustrate this reduction for the 1L-AFA $(A = \{q_0, q_1, q_2\}, \delta, q_0, \{q_2\})$ with $\delta(q_0) = q_1 \wedge q_2$ and $\delta(q_1) = q_1 \vee q_2$ and $\delta(q_2) = q_1 \vee q_1$. Here is a visualization of A:



Figures 7 and 8 show parts of the pLTS generated by the pOCA obtained by applying the reduction from the proof of Theorem 5.4 to A. It can be seen in Figure 7 that $q_1 Z \sim q_1' Z$ and $q_2 Z \not\sim q_2' Z$, reflecting the facts that $\neg Acc(q_1, 0)$ and $Acc(q_2, 0)$. Therefore $r_1 IZ \sim r_1' IZ$ and $r_2 IZ \not\sim r_2' IZ$. Corresponding to the transition $\delta(q_1) = q_1 \vee q_2$, Figure 7 shows an AND-gadget, see Figure 6 (left). Thus we have $q_1 IZ \not\sim q_1' IZ$, reflecting the fact that $Acc(q_1, 1)$.

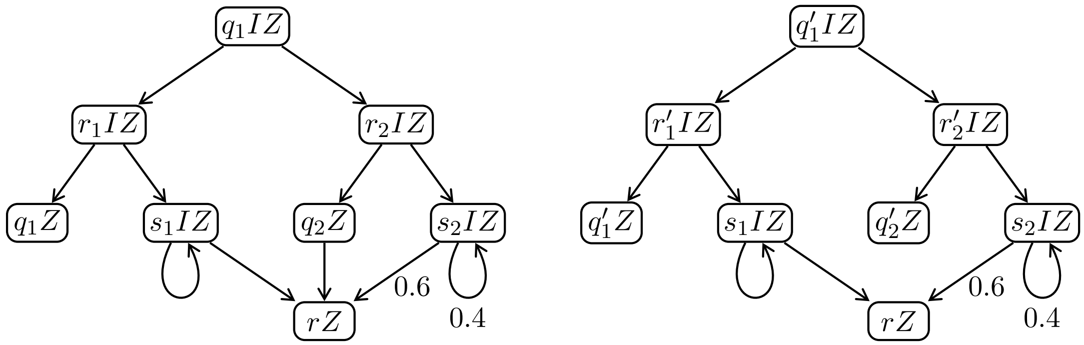


Figure 7: A part of the pLTS generated by the pOCA obtained by applying the reduction from the proof of Theorem 5.4 to the 1L-AFA A from Example 5.5. Unless indicated otherwise, for each state there is a uniform distribution on the outgoing transitions. For better readability, some states appear twice.

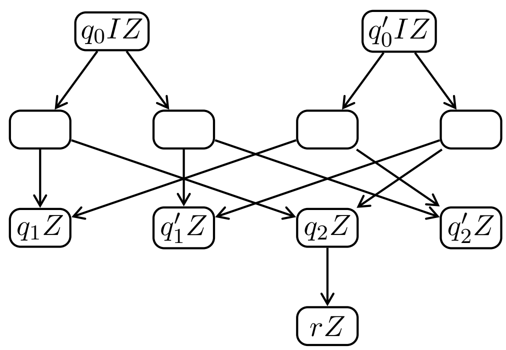


Figure 8: A part of the pLTS generated by the pOCA obtained by applying the reduction from the proof of Theorem 5.4 to the 1L-AFA A from Example 5.5. For each state there is a uniform distribution on the outgoing transitions.

In Figure 8 it can be seen again that $q_1Z \sim q'_1Z$ and $q_2Z \not\sim q'_2Z$, reflecting the facts that $\neg Acc(q_1, 0)$ and $Acc(q_2, 0)$. Corresponding to the transition $\delta(q_0) = q_1 \wedge q_2$, Figure 7 shows an OR-gadget, see Figure 6 (right). Thus we have $q_0IZ \sim q'_0IZ$, reflecting the fact that $\neg Acc(q_0, 1)$.

5.2. **Bisimilarity of pvPDA is EXPTIME-hard.** In this section we prove the following:

Theorem 5.6. *Bisimilarity is EXPTIME-hard (even) for fully probabilistic pvPDA* $\Delta = (Q, \Gamma, \Sigma, \hookrightarrow)$ *with* $|\Sigma_r| = |\Sigma_{int}| = |\Sigma_c| = 1$.

It was shown in [Srb09] that bisimilarity for (non-probabilistic) vPDA is EXPTIME-complete. The hardness result there follows by observing that the proof given in [KM02] (see also [KM10]) for general PDA works even for vPDA. Referring to [KM02], it is commented in [Srb09]: "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 [Wal01]. Our reduction proves Theorem 5.6 and at the same time provides an alternative proof for (standard) vPDA.

Proof of Theorem 5.6. Let $\Delta = (Q, \Gamma, \{a\}, \hookrightarrow)$ be a unary PDA (the actions do not matter in reachability games) with a control state partition $Q = Q_0 \cup Q_1$ and an initial configuration p_0X_0. We call a configuration $pX\alpha$ *dead* if it has no successor configuration, i.e., if Δ 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 $\mathcal{L}_\Delta$: First the current configuration is p_0X_0; in a current configuration $p\alpha$ with $p \in Q_i$ (where $i \in \{0,1\}$), Player i chooses a successor configuration of $p\alpha$ in $\mathcal{L}_\Delta$ as a new current configuration. The goal of Player 1 is to reach a dead configuration; the goal of Player 0 is to avoid that. It is shown in [Wal01, pp. 261–262] that determining the winner in that game is EXPTIME-hard.

W.l.o.g. we assume that for each $pX \in Q\Gamma$ there are *at most* two rules in $\hookrightarrow$ with pX on the left-hand side; moreover, if they are two, then they are of the form $pX \stackrel{a}{\hookrightarrow} p_1X_1$ and $pX \stackrel{a}{\hookrightarrow} p_2X_2$ (for $X_1, X_2 \in \Gamma$). We further assume that no configuration with the empty stack is reachable from p_0X_0.

We will construct a fully probabilistic vPDA $\bar{\Delta} = (\bar{Q}, \Gamma, \{a_r, a_{int}, a_c\}, \circ\!\!\rightarrow)$ such that for each control state $p \in Q$ the set $\bar{Q}$ includes p and a copy p', and the configurations p_0X_0 and p'_0X_0 of $\bar{\Delta}$ are bisimilar if and only if Player 0 has a winning strategy (in the reachability game from p_0X_0):

- For each $pX \in Q\Gamma$ that is dead in Δ, in $\bar{\Delta}$ we create a rule $pX \stackrel{a_{int}}{\circ\!\!\longrightarrow} pX$ and a rule $p'X \stackrel{a_{int}}{\circ\!\!\longrightarrow} zX$ 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 Δ (and hence Player 1 wins), then we have $pX \not\sim p'X$ in $\mathcal{L}_{\bar{\Delta}}$.
- For each rule $pX \stackrel{a}{\hookrightarrow} q\alpha$ such that pX is not the left-hand side of any other rule in Δ, in $\bar{\Delta}$ we create rules $pX \stackrel{a}{\circ\!\!\rightarrow} q\alpha$ and $p'X \stackrel{a}{\circ\!\!\rightarrow} q'\alpha$, where $a = a_r, a_{int}, a_c$ if $|\alpha| = 0, 1, 2$, respectively.
- For each pair of (different) rules $pX \stackrel{a}{\hookrightarrow} p_1X_1$, $pX \stackrel{a}{\hookrightarrow} p_2X_2$:
  - If $p \in Q_0$, then we implement an OR-gadget from Figure 6 (right): let $(p_1X_1p_2X_2), (p'_1X_1p'_2X_2), (p_1X_1p'_2X_2), (p'_1X_1p_2X_2) \in \bar{Q}$ be fresh control states, and add rules
    - $pX \circ\!\!\rightarrow (p_1X_1p_2X_2)X \mid (p'_1X_1p'_2X_2)X$ (this is a shorthand for $pX \stackrel{a_{int}}{\circ\!\!\longrightarrow} 0.5(p_1X_1p_2X_2)X + 0.5(p'_1X_1p'_2X_2)X$),
    - $p'X \circ\!\!\rightarrow (p_1X_1p'_2X_2)X \mid (p'_1X_1p_2X_2)X$,
    - $(p_1X_1p_2X_2)X \circ\!\!\rightarrow p_1X_1 \mid p_2X_2$, $(p'_1X_1p'_2X_2)X \circ\!\!\rightarrow p'_1X_1 \mid p'_2X_2$, $(p_1X_1p'_2X_2)X \circ\!\!\rightarrow p_1X_1 \mid p'_2X_2$, $(p'_1X_1p_2X_2)X \circ\!\!\rightarrow p'_1X_1 \mid p_2X_2$.
  - If $p \in Q_1$ we implement an AND-gadget from Figure 6 (left): let $(p_1X_1), (p'_1X_1), (p_2X_2), (p'_2X_2) \in \bar{Q}$ be fresh control states, and add rules
    - $pX \circ\!\!\rightarrow (p_1X_1)X \mid (p_2X_2)X$ and $p'X \circ\!\!\rightarrow (p'_1X_1)X \mid (p'_2X_2)X$,
    - $(p_1X_1)X \stackrel{a_{int}}{\circ\!\!\longrightarrow} p_1X_1$ and $(p'_1X_1)X \stackrel{a_{int}}{\circ\!\!\longrightarrow} p'_1X_1$,
    - $(p_2X_2)X \circ\!\!\rightarrow p_2X_2 \mid zX$ and $(p'_2X_2)X \circ\!\!\rightarrow p'_2X_2 \mid zX$.

    Here, the transitions to zX serve to implement the condition $t_1 \not\sim t'_2$ from Proposition 5.2.

An induction argument now easily establishes that $p_0X_0 \sim p'_0X_0$ holds in $\bar{\Delta}$ if and only if Player 0 has a winning strategy in the game in Δ.

We note that the same reduction works for non-probabilistic vPDA, if the probabilistic branching is replaced by non-deterministic branching; we thus get the mentioned alternative proof of EXPTIME-hardness for standard vPDA. □

6. Conclusion

There are a number of variants of standard (non-deterministic) pushdown automata for which the problem of checking bisimilarity is elementary. For three of the most prominent such classes—one-counter automata, visibly pushdown automata, and basic process algebra—we have shown that checking bisimilarity for the probabilistic extensions incurs no cost in computational complexity over the standard case. More precisely, for the respective probabilistic extensions of these three models we recover the same complexity upper bounds for checking bisimilarity as in the standard case. Thus the message of this paper is that adding probability comes at no extra cost for checking bisimilarity of pushdown automata.

At a technical level, the paper is constructed around a simple equivalence-preserving transformation that eliminates probabilistic transitions. However since this transformation incurs an exponential blow-up when used as a black-box, we have had to resort to bespoke arguments for each subclass of pPDA in order to obtain optimal complexity bounds from our basic underlying reduction. We also note that hitherto rather bespoke proofs were used in this area [BKS08, FK11]. Thus a natural question arising from this work is whether one can identify general conditions on a class of pPDA that enable this "bisimilarity reduction" to go through without incurring an exponential blow-up.

A second main message of this paper is that checking bisimilarity for (subclasses of) PDA is no easier in the fully probabilistic case than in the standard case. This contrasts with the situation for language equivalence (e.g., deciding language equivalence of nondeterministic finite automata is PSPACE-complete, whereas the natural analog of language equivalence for fully probabilistic finite automata is decidable in polynomial time [Tze92]). In light of this, another interesting question is whether language equivalence of fully probabilistic PDA (without ε-transitions) is decidable. This question is currently open to the best of our knowledge (and related to other problems in language theory [FJKW14]).

Acknowledgements. Vojtěch Forejt was at Oxford University when most of the research was carried out. Petr Jančar was at Techn. Univ. Ostrava, supported by the grant GAČR:15-13784S (finishing in 2017) of the Grant Agency of the Czech Rep. Stefan Kiefer is supported by the Royal Society.

References

[AW06] S. Andova and T. Willemse. Branching bisimulation for probabilistic systems: Characteristics and decidability. *Theor. Comput. Sci.*, 356(3):325–355, 2006.

[Bai96] C. Baier. Polynomial time algorithms for testing probabilistic bisimulation and simulation. In *CAV (Computer Aided Verification)*, volume 1102 of *LNCS*, pages 50–61. Springer, 1996.

[BCMS01] 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.

[BEM00] C. Baier, B. Engelen, and M.E. Majster-Cederbaum. Deciding bisimilarity and similarity for probabilistic processes. *J. Comput. Syst. Sci.*, 60(1):187–231, 2000.

[BGJ10] S. Böhm, S. Göller, and P. Jančar. Bisimilarity of one-counter processes is PSPACE-complete. In *CONCUR*, volume 6269 of *LNCS*, pages 177–191. Springer, 2010.

[BGJ14] S. Böhm, S. Göller, and P. Jančar. Bisimulation equivalence and regularity for real-time one-counter automata. *J. Comput. Syst. Sci.*, 80(4):720–743, 2014.

[BGKM13] M. Benedikt, S. Göller, S. Kiefer, and A.S. Murawski. Bisimilarity of pushdown automata is nonelementary. In *LICS (Logic in Computer Science)*, pages 488–498. IEEE Computer Society, 2013.

[BH97] C. Baier and H. Hermanns. Weak bisimulation for fully probabilistic processes. In *CAV*, volume 1254 of *LNCS*, pages 119–130. Springer, 1997.

[BKS08] T. Brázdil, A. Kučera, and O. Stražovský. Deciding probabilistic bisimilarity over infinite-state probabilistic systems. *Acta Inf.*, 45(2):131–154, 2008.

[CvBW12] D. Chen, F. van Breugel, and J. Worrell. On the complexity of computing probabilistic bisimilarity. In *FoSSaCS*, volume 7213 of *LNCS*, pages 437–451. Springer, 2012.

[EKM06] J. Esparza, A. Kučera, and R. Mayr. Model checking probabilistic pushdown automata. *Logical Methods in Computer Science*, 2(1), 2006.

[EWY10] 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.

[EY09] K. Etessami and M. Yannakakis. Recursive Markov chains, stochastic grammars, and monotone systems of nonlinear equations. *J. ACM*, 56(1):1:1–1:66, 2009.

[FJKW12] V. Forejt, P. Jančar, S. Kiefer, and J. Worrell. Bisimilarity of probabilistic pushdown automata. In *FSTTCS*, volume 18 of *LIPIcs*, pages 448–460. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2012.

[FJKW14] V. Forejt, P. Jančar, S. Kiefer, and J. Worrell. Language equivalence of probabilistic pushdown automata. *Information and Computation*, 237:1–11, 2014.

[FK11] H. Fu and J.-P. Katoen. Deciding probabilistic simulation between probabilistic pushdown automata and finite-state systems. In *FSTTCS*, volume 13 of *LIPIcs*, pages 445–456. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2011.

[Hol95] M. Holzer. On emptiness and counting for alternating finite automata. In *DLT (Developments in Language Theory)*, pages 88–97, 1995.

[Jan13] P. Jančar. Bisimilarity on basic process algebra is in 2-ExpTime (an explicit proof). *Logical Methods in Computer Science*, 9(1), 2013.

[Jan14a] P. Jančar. Bisimulation equivalence of first-order grammars. In *ICALP (Part II)*, volume 8573 of *LNCS*, pages 232–243. Springer, 2014.

[Jan14b] P. Jančar. Equivalences of pushdown systems are hard. In *FoSSaCS*, volume 8412 of *LNCS*, pages 1–28. Springer, 2014.

[JS07] 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.

[Kie13] S. Kiefer. BPA bisimilarity is EXPTIME-hard. *Inf. Process. Lett.*, 113(4):101–106, 2013.

[KM02] A. Kučera and R. Mayr. On the complexity of semantic equivalences for pushdown automata and BPA. In *MFCS*, volume 2420 of *LNCS*, pages 433–445. Springer, 2002.

[KM10] 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.

[May03] R. Mayr. Undecidability of weak bisimulation equivalence for 1-counter processes. In *ICALP*, volume 2719 of *LNCS*, pages 570–583. Springer, 2003.

[PLS00] A. Philippou, I. Lee, and O. Sokolsky. Weak bisimulation for probabilistic systems. In *CONCUR*, volume 1877 of *LNCS*, pages 334–349. Springer, 2000.

[Sén05] G. Sénizergues. The bisimulation problem for equational graphs of finite out-degree. *SIAM J. Comput.*, 34(5):1025–1106, 2005.

[SL94] R. Segala and N. A. Lynch. Probabilistic simulations for probabilistic processes. In *CONCUR*, volume 836 of *LNCS*, pages 481–496. Springer, 1994.

[Srb04] J. Srba. *Roadmap of Infinite Results*, volume Vol 2: Formal Models and Semantics. World Scientific Publishing Co., 2004. (Cf. `http://people.cs.aau.dk/~srba/roadmap/`).

[Srb09] J. Srba. Beyond language equivalence on visibly pushdown automata. *Logical Methods in Computer Science*, 5(1):2, 2009.

[Tze92] W. Tzeng. A polynomial-time algorithm for the equivalence of probabilistic automata. *SIAM J. Comput.*, 21(2):216–227, 1992.

[Wal01] I. Walukiewicz. Pushdown processes: Games and model-checking. *Information and Computation*, 164(2):234–263, 2001.

[ZYS$^+$18] L. Zhang, P. Yang, L. Song, H. Hermanns, C. Eisentraut, D.N. Jansen, and J.C. Godskesen. Probabilistic bisimulation for realistic schedulers. *Acta Informatica*, 55(6):461–488, 2018.