Model Checking Probabilistic Pushdown Automata

Javier Esparza
Institute for Formal Methods in Computer Science,
University of Stuttgart,
Universität str. 38, 70569 Stuttgart, Germany.
esparza@informatik.uni-stuttgart.de

Antonín Kučera*
Faculty of Informatics, Masaryk University,
Botanická 68a, 60200 Brno,
Czech Republic.
tony@fi.muni.cz

Richard Mayr†
Department of Computer Science,
Albert-Ludwigs-University Freiburg
Georges-Koehler-Allee 51,
D-79110 Freiburg, Germany.
mayrri@informatik.uni-freiburg.de

Abstract

We consider the model checking problem for probabilistic pushdown automata (pPDA) and properties expressible in various probabilistic logics. We start with properties that can be formulated as instances of a generalized random walk problem. We prove that both qualitative and quantitative model checking for this class of properties and pPDA is decidable. Then we show that model checking for the qualitative fragment of the logic PCTL and pPDA is also decidable. Moreover, we develop an error-tolerant model checking algorithm for general PCTL and the subclass of stateless pPDA. Finally, we consider the class of properties definable by deterministic Büchi automata, and show that both qualitative and quantitative model checking for pPDA is decidable.

1. Introduction

Probabilistic systems can be used for modeling systems that exhibit uncertainty, such as communication protocols over unreliable channels, randomized distributed systems, or fault-tolerant systems. Finite-state models of such systems often use variants of probabilistic automata whose underlying semantics is defined in terms of homogeneous Markov chains, which are also called "fully probabilistic transition systems" in this context. For fully probabilistic finite-state systems, algorithms for various (probabilistic) temporal logics like LTL, PCTL, PCTL*, probabilistic μ-calculus, etc., have been presented in [22, 18, 26, 10, 17, 5, 11, 19, 12]. As for infinite-state systems, most works so far considered probabilistic lossy channel systems [20] which model asynchronous communication through unreliable channels [7, 1, 2, 8]. A notable recent result is the decidability of quantitative model checking of liveness properties specified by Büchi-automata for probabilistic lossy channel systems [24]. In fact, this algorithm is *error tolerant* in the sense that the quantitative model checking is solved only up to an arbitrarily small (but non-zero) given error.

In this paper we consider *probabilistic pushdown automata (pPDA)*, which are a natural model for probabilistic sequential programs with recursive procedure calls. There is a large number of results about model checking of non-probabilistic PDA or similar models (see for instance [4, 9, 13, 27]), but the probabilistic extension has so far not been considered. As a related work we can mention [23], where it is shown that a restricted subclass of pPDA (where essentially all probabilities for outgoing arcs are either 1 or $1/2$) generates a richer class of languages than non-deterministic PDA. Another work [3] shows the equivalence of pPDA and probabilistic context-free grammars.

Here we consider model checking problems for pPDA (and its natural subclass of *stateless pPDA* denoted pBPA[1])

* On leave at the Institute for Formal Methods in Computer Science, University of Stuttgart. Supported by the Alexander von Humboldt Foundation and by the Grant Agency of the Czech Republic, grant No. 201/03/1161.

† Supported by Landesstiftung Baden–Württemberg, grant No. 21–655.023.

1 This is a standard notation adopted in concurrency theory. The sub-

and various probabilistic logics. We start with a class of

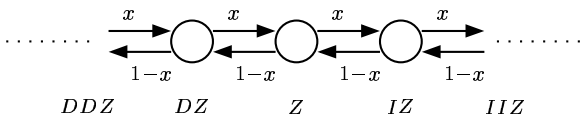

Figure 1. Bernoulli random walk as a pBPA

properties that can be specified as a generalized *random walk problem*. To get a better intuition about this class of problems, realize that some random walks can easily be specified by pBPA systems. For example, consider a pBPA with just three stack symbols Z, I, D and transitions $Z \xrightarrow{x} IZ$, $Z \xrightarrow{1-x} DZ$, $I \xrightarrow{x} II$, $I \xrightarrow{1-x} \varepsilon$, $D \xrightarrow{1-x} DD$, and $D \xrightarrow{x} \varepsilon$, where $x \in [0,1]$. Then the transition graph of Z (see Fig. 1) is the well-known *Bernoulli walk*. A typical question examined in theory of random walks is "Do we eventually revisit a given state (with probability one)?", or more generally "What is the probability of reaching a given state from another given state?" For example, it is a standard result that the state Z of Fig. 1 is revisited with probability 1 iff $x = 1/2$. This simple example indicates that answers to qualitative questions about pPDA (i.e., whether something holds with probability 1 or 0) depend on the exact probabilities of individual transitions. This is different from finite-state systems where qualitative properties depend only on the topology of a given finite-state Markov chain.

The generalized random walk problem is formulated as follows: Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be subsets of the set of states of a given Markov chain, and let s be a state of $\mathcal{C}_1$. What is the probability that, starting at s, a state of $\mathcal{C}_2$ is reached via a path leading only through states of $\mathcal{C}_1$? Let us denote this probability by $\mathcal{P}(s, \mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2)$. The problem of computing $\mathcal{P}(s, \mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2)$ has been previously considered (and solved) for finite-state systems, where this probability can be computed precisely [17, 11]. In Section 3, we propose a solution for pPDA applicable to those sets $\mathcal{C}_1, \mathcal{C}_2$ which are *regular*, i.e., recognizable by finite-state automata. More precisely, we show that the problem whether $\mathcal{P}(s, \mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2) \sim \varrho$, where $\sim \in \{\leq, <, \geq, >, =\}$ and $\varrho \in [0,1]$, is decidable. Interestingly, this is achieved without explicitly computing the probability $\mathcal{P}(s, \mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2)$. Nevertheless, for an arbitrary precision $0 < \lambda < 1$ we can compute rational lower and upper approximations $\mathcal{P}^\ell, \mathcal{P}^u \in [0,1]$ such that $\mathcal{P}^\ell \leq \mathcal{P}(s, \mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2) \leq \mathcal{P}^u$ and $\mathcal{P}^u - \mathcal{P}^\ell \leq \lambda$.

In Section 4, we consider the model checking problem for pPDA and the logic PCTL. This is a more general problem than the one about random walks (the class of properties expressible in PCTL is strictly larger). In Section 4.1, we give a model checking algorithm for the *qualitative fragment* of PCTL and pPDA processes. For general PCTL formulae and pBPA processes, an *error tolerant* model checking algorithm is developed in Section 4.2. The question whether this result can be extended to pPDA is left open.

Finally, in Section 5 we prove that both qualitative and quantitative model checking for the class of properties definable by deterministic Büchi automata is decidable for pPDA. Again, this is done without computing the probability explicitly, but rational lower and upper approximations can be computed up to an arbitrarily small given error.

Due to the lack of space, some proofs have been omitted. These can be found in a full version of this paper [14].

2. Preliminary Definitions

Definition 2.1. *A (fully)* probabilistic transition system *is a triple $\mathcal{T} = (S, \to, Prob)$ where S is a finite or countably infinite set of* states, *$\to \subseteq S \times S$ is a* transition relation, *and $Prob$ is a function which to each transition $s \to t$ of $\mathcal{T}$ assigns its probability $Prob(s \to t) \in (0,1]$ so that for every $s \in S$ we have $\sum_{s \to t} Prob(s \to t) \in \{0, 1\}$. (The sum above can be 0 if it is empty, i.e., if s does not have any outgoing transitions.)*

In the rest of this paper we also write $s \xrightarrow{x} t$ instead of $Prob(s \to t) = x$. A *path* in $\mathcal{T}$ is a finite or infinite sequence $w = s_0, s_1, \cdots$ of states such that $s_i \to s_{i+1}$ for every i. We also use $w(i)$ to denote the state s_i of w (by writing $w(i) = s$ we implicitly impose the condition that the length of w is at least $i + 1$). A *run* is a maximal path, i.e., a path which cannot be prolonged. The sets of all finite paths and all runs of $\mathcal{T}$ are denoted $FPath$ and Run, respectively[2]. Similarly, the sets of all finite paths and runs that start in a given $s \in S$ are denoted $FPath(s)$ and $Run(s)$, respectively.

Each $w \in FPath$ determines a *basic cylinder* $Run(w)$ which consists of all runs that start with w. To every $s \in S$ we associate the probabilistic space $(Run(s), \mathcal{F}, \mathcal{P})$ where $\mathcal{F}$ is the σ-field generated by all basic cylinders $Run(w)$ where w starts with s, and $\mathcal{P} : \mathcal{F} \to [0,1]$ is the unique probability function such that $\mathcal{P}(Run(w)) = \Pi_{i=0}^{m-1} x_i$ where $w = s_0, \cdots, s_m$ and $s_i \xrightarrow{x_i} s_{i+1}$ for every $0 \leq i < m$ (if $m = 0$, we put $\mathcal{P}(Run(w)) = 1$). We say that sets $A \subseteq FPath(s)$ and $A' \subseteq FPath(s')$ are $\mathcal{P}$*-equivalent* iff $\sum_{w \in A} \mathcal{P}(Run(w)) = \sum_{w \in A'} \mathcal{P}(Run(w))$.

The Logic PCTL

PCTL, the probabilistic extension of CTL, was defined by Hansson & Jonsson in [17]. Let $Ap = \{a, b, c, \dots\}$ be a

class of stateless PDA corresponds to a natural subclass of ACP known as Basic Process Algebra [6].

2 In this paper, the $\mathcal{T}$ is always clear from the context.

countably infinite set of *atomic propositions*. The syntax of PCTL[3] is given by the following abstract syntax equation:

$$\varphi \quad ::= \quad \mathtt{tt} \mid a \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \mathcal{X}^{\sim\varrho}\varphi \mid \varphi_1 \,\mathcal{U}^{\sim\varrho}\, \varphi_2$$

Here a ranges over Ap, $\varrho \in [0,1]$, and $\sim \in \{\leq, <, \geq, >\}$. Let $\mathcal{T} = (S, \to, Prob)$ be a probabilistic transition system. For all $s \in S$, all $\mathcal{C}, \mathcal{C}_1, \mathcal{C}_2 \subseteq S$, and all $k \in \mathbb{N}_0$, let

- $Run(s, \mathcal{X}\mathcal{C}) = \{w \in Run(s) \mid w(1) \in \mathcal{C}\}$
- $Run(s, \mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2) = \{w \in Run(s) \mid \exists i \geq 0 : w(i) \in \mathcal{C}_2 \text{ and } w(j) \in \mathcal{C}_1 \text{ for all } 0 \leq j < i\}$
- $FPath^k(s, \mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2) = \{s_0, \cdots, s_\ell \in FPath(s) \mid 0 \leq \ell \leq k, s_\ell \in \mathcal{C}_2 \text{ and } s_j \in \mathcal{C}_1 \smallsetminus \mathcal{C}_2 \text{ for all } 0 \leq j < \ell\}$
- $FPath(s, \mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2) = \bigcup_{k=0}^{\infty} FPath^k(s, \mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2)$

Obviously,

$$\mathcal{P}(Run(s, \mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2)) = \sum_{w \in FPath(s, \mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2)} \mathcal{P}(Run(w)).$$

Let $\nu : Ap \to 2^S$ be a *valuation*. The denotation of a PCTL formula φ over $\mathcal{T}$ w.r.t. ν, denoted $[\![\varphi]\!]^\nu$, is defined inductively as follows:

$$\begin{array}{rcl}
[\![\mathtt{tt}]\!]^\nu & = & S \\
[\![a]\!]^\nu & = & \nu(a) \\
[\![\neg\varphi]\!]^\nu & = & S \smallsetminus [\![\varphi]\!]^\nu \\
[\![\varphi_1 \wedge \varphi_2]\!]^\nu & = & [\![\varphi_1]\!]^\nu \cap [\![\varphi_2]\!]^\nu \\
[\![\mathcal{X}^{\sim\varrho}\varphi]\!]^\nu & = & \{s \in S \mid \mathcal{P}(Run(s, \mathcal{X}[\![\varphi]\!]^\nu)) \sim \varrho\} \\
[\![\varphi_1 \,\mathcal{U}^{\sim\varrho}\, \varphi_2]\!]^\nu & = & \{s \in S \mid \mathcal{P}(Run(s, [\![\varphi_1]\!]^\nu \,\mathcal{U}\, [\![\varphi_2]\!]^\nu)) \sim \varrho\}
\end{array}$$

As usual, we write $s \models^\nu \varphi$ instead of $s \in [\![\varphi]\!]^\nu$.

The *qualitative fragment* of PCTL is obtained by restricting the allowed operator/number combinations to '≤ 0' and '≥ 1', which will be also written as '$= 0$' and '$= 1$', resp. (Observe that '< 1', '> 0' are definable from '≤ 0', '≥ 1', and negation; for example, $a \,\mathcal{U}^{<1} b \equiv \neg(a \,\mathcal{U}^{\geq 1} b)$.)

Probabilistic PDA

Definition 2.2. *A* probabilistic pushdown automaton (pPDA) *is a tuple* $\Delta = (Q, \Gamma, \delta, Prob)$ *where* Q *is a finite set of* control states, Γ *is a finite* stack alphabet, $\delta \subseteq Q \times \Gamma \times Q \times \Gamma^*$ *is a finite* transition relation *(we write* $pX \to q\alpha$ *instead of* $(p, X, q, \alpha) \in \delta$*), and* $Prob$ *is a function which to each transition* $pX \to q\alpha$ *assigns its probability* $Prob(pX \to q\alpha) \in (0, 1]$ *so that for all* $p \in Q$ *and* $X \in \Gamma$ *we have that* $\sum_{pX \to q\alpha} Prob(pX \to q\alpha) \in \{0, 1\}$.

A pBPA *is a pPDA with just one control state. Formally, a pBPA is understood as a triple* $\Delta = (\Gamma, \delta, Prob)$ *where* $\delta \subseteq \Gamma \times \Gamma^*$.

In the rest of this paper we adopt a more intuitive notation, writing $pX \xrightarrow{x} q\alpha$ instead of $Prob(pX \to q\alpha) = x$.

3 For simplicity we omit the bounded 'until' operator of [17].

The set $Q \times \Gamma^*$ of all configurations of Δ is denoted by $\mathcal{C}(\Delta)$. We also assume (w.l.o.g.) that if $pX \to q\alpha \in \delta$, then $|\alpha| \leq 2$.

To Δ we associate the probabilistic transition system $\mathcal{T}_\Delta$ where $\mathcal{C}(\Delta)$ is the set of states and the probabilistic transition relation is determined by $pX\beta \xrightarrow{x} q\alpha\beta$ iff $pX \xrightarrow{x} q\alpha$.

The model checking problem for pPDA configurations and PCTL formulae (i.e., the question whether $p\alpha \models^\nu \varphi$ for given $p\alpha$, φ, and ν) is clearly undecidable for general valuations. Therefore, we restrict ourselves to *regular* valuations which to every $a \in Ap$ assign a *regular set of configurations*:

Definition 2.3. *A* Δ-automaton *is a triple* $\mathcal{A} = (St, \gamma, Acc)$ *where* St *is a finite set of* states *s.t.* $Q \subseteq St$, $\gamma : St \times \Gamma \to St$ *is a (total)* transition function, *and* $Acc \subseteq St$ *a set of* accepting states.

The function γ *is extended to the elements of* Γ^* *in the standard way. Each* Δ*-automaton* $\mathcal{A}$ *determines a set* $\mathcal{C}(\mathcal{A}) \subseteq \mathcal{C}(\Delta)$ *given by* $p\alpha \in \mathcal{C}(\mathcal{A})$ *iff* $\gamma(p, \alpha^R) \in Acc$*. Here* α^R *is the reverse of* α*, i.e., the word obtained by reading* α *from right to left.*

We say that a set $\mathcal{C} \subseteq \mathcal{C}(\Delta)$ *is* regular *iff there is a* Δ*-automaton* $\mathcal{A}$ *such that* $\mathcal{C} = \mathcal{C}(\mathcal{A})$.

In other words, regular sets of configurations are recognizable by finite-state automata which read the stack bottom-up (the bottom-up direction was chosen just for technical convenience).

3. Random Walks on pPDA Graphs

For the rest of this section, let us fix a pPDA $\Delta = (Q, \Gamma, \delta, Prob)$.

An important technical step in our development is the replacement of regular sets of configurations with "simple" ones for which the membership function depends just on the control state and the top stack symbol of a given configuration.

Definition 3.1. *A set of configurations* $\mathcal{C}$ *is* simple *if there is a set* $G \subseteq Q \times (\Gamma \cup \{\varepsilon\})$ *such that for each* $p\alpha \in \mathcal{C}(\Delta)$ *we have that* $p\alpha \in \mathcal{C}$ *iff either* $\alpha = \varepsilon$ *and* $p\varepsilon \in G$*, or* $\alpha = X\beta$ *and* $pX \in G$.

The next lemma says that regular sets of configurations can be effectively replaced with simple ones. A proof is standard (see, e.g.,[15]).

Lemma 3.2. *For each pPDA* $\Delta = (Q, \Gamma, \delta, Prob)$ *and regular sets* $\mathcal{C}_1, \cdots, \mathcal{C}_k \subseteq \mathcal{C}(\Delta)$ *there effectively exist a pPDA* $\Delta' = (Q, \Gamma', \delta', Prob')$*, simple sets* $\mathcal{C}'_1, \cdots, \mathcal{C}'_k \subseteq \mathcal{C}(\Delta')$*, and an injective mapping* $\mathcal{G} : \mathcal{C}(\Delta) \to \mathcal{C}(\Delta')$ *such that for each* $p\alpha \in \mathcal{C}(\Delta)$ *the following conditions are satisfied:*

- *if* $p\alpha \xrightarrow{x} q\beta$*, then* $\mathcal{G}(p\alpha) \xrightarrow{x} \mathcal{G}(q\beta)$*;*

- *if* $\mathcal{G}(p\alpha) \xrightarrow{x} s$ *for some* $s \in \mathcal{C}(\Delta')$, *then there is* $p\alpha \xrightarrow{x} q\beta$ *such that* $\mathcal{G}(q\beta) = s$;
- *for each* $1 \leq j \leq k$ *we have* $p\alpha \in \mathcal{C}_j$ *iff* $\mathcal{G}(p\alpha) \in \mathcal{C}'_j$.

Moreover, if $\mathcal{C} \subseteq \mathcal{C}(\Delta')$ *is regular, then* $\mathcal{G}^{-1}(\mathcal{C})$ *is also regular.*

For the rest of this section, let $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathcal{C}(\Delta)$ be (fixed) simple sets, and let $G_1, G_2 \subseteq Q \times (\Gamma \cup \{\varepsilon\})$ be the sets associated to $\mathcal{C}_1, \mathcal{C}_2$ in the sense of Definition 3.1. To simplify our notation, we adopt the following conventions:

- For each $\mathcal{C} \subseteq \mathcal{C}(\Delta)$, let $\mathcal{C}^\bullet = \mathcal{C} \setminus (Q \times \{\varepsilon\})$. Observe that if $\mathcal{C}$ is simple, then so is $\mathcal{C}^\bullet$.
- For all $p, q \in Q$ and $X \in \Gamma$, we use $[pXq]$ to abbreviate $\mathcal{P}(pX, \mathcal{C}_1 \setminus \mathcal{C}_2 \,\mathcal{U}\, \{q\varepsilon\})$, and $[pX\bullet]$ to abbreviate $\mathcal{P}(pX, \mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2^\bullet)$.

The next lemma says how to compute $\mathcal{P}(Run(pX_1 \cdots X_n, \mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2))$ from the finite family of all $[pXq]$, $[pX\bullet]$ probabilities.

Lemma 3.3. *For each* $pX_1 \cdots X_n \in \mathcal{C}(\Delta)$ *where* $n \geq 0$ *we have that* $\mathcal{P}(Run(pX_1 \cdots X_n, \mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2))$ *equals to*

$$\sum_{i=1}^{n} \sum_{\substack{(q_1,\cdots,q_i)\in Q^i \\ \text{where } p=q_1}} [q_i X_i \bullet] \cdot \prod_{j=1}^{i-1} [q_j X_j q_{j+1}] \;+\; \sum_{\substack{(q_1,\cdots,q_{n+1})\in Q^{n+1} \\ \text{where } p=q_1 \text{ and } q_{n+1}\varepsilon \in \mathcal{C}_2}} \prod_{j=1}^{n} [q_j X_j q_{j+1}]$$

with the convention that empty sum equals to 0 *and empty product equals to* 1.

Now we show that the probabilities $[pXq]$, $[pX\bullet]$ form the least solution of an effectively constructible system of equations. This can be seen as a generalization of a similar result for finite-state systems [17, 11]. In the finite-state case, the equations are linear and can be further modified so that they have a *unique* solution (which is then computable, e.g., by Gauss elimination). In the case of pPDA, the equations are not linear and cannot be generally solved by analytical methods. The question whether the equations can be further modified so that they have a unique solution is left open; we just note that the method used for finite-state systems is insufficient (this is demonstrated by Example 3.5).

Let $\mathcal{V} = \{\langle pXq\rangle, \langle pX\bullet\rangle \mid p, q \in Q, X \in \Gamma\}$ be a set of "variables". Let us consider the system of recursive equations constructed as follows:

- if $pX \notin G_1 \setminus G_2$, then $\langle pXq\rangle = 0$ for each $q \in Q$; otherwise, we put

$$\langle pXq\rangle \;=\; \sum_{pX \xrightarrow{x} rYZ} x \cdot \sum_{t\in Q} \langle rYt\rangle \cdot \langle tZq\rangle \;+\; \sum_{pX \xrightarrow{x} rY} x \cdot \langle rYq\rangle \;+\; \sum_{pX \xrightarrow{x} q\varepsilon} x$$

- if $pX \in G_2$, then $\langle pX\bullet\rangle = 1$; if $pX \notin G_1 \cup G_2$, then $\langle pX\bullet\rangle = 0$; otherwise we put

$$\langle pX\bullet\rangle \;=\; \sum_{pX \xrightarrow{x} rYZ} x \cdot (\langle rY\bullet\rangle + \sum_{t\in Q} \langle rYt\rangle \cdot \langle tZ\bullet\rangle) \;+\; \sum_{pX \xrightarrow{x} rY} x \cdot \langle rY\bullet\rangle$$

For given $t \in [0,1]^{|\mathcal{V}|}$, $p, q \in Q$, and $X \in \Gamma$ we use $\langle pXq\rangle_t$ and $\langle pX\bullet\rangle_t$ to denote the component of t which corresponds to the variable $\langle pXq\rangle$ and $\langle pX\bullet\rangle$, respectively. The above defined system of equations determines a unique operator $\mathcal{F} : [0,1]^{|\mathcal{V}|} \to [0,1]^{|\mathcal{V}|}$ where $\mathcal{F}(t)$ is the tuple of values obtained by evaluating the right-hand sides of the equations where all $\langle pXq\rangle$ and $\langle pX\bullet\rangle$ are substituted with $\langle pXq\rangle_t$ and $\langle pX\bullet\rangle_t$, respectively.

Theorem 3.4. *The operator* $\mathcal{F}$ *has the least fixed-point* μ. *Moreover, for all* $p, q \in Q$ *and* $X \in \Gamma$ *we have that* $\langle pXq\rangle_\mu = [pXq]$ *and* $\langle pX\bullet\rangle_\mu = [pX\bullet]$.

Example 3.5. *Let us consider the pBPA system* Δ *of Fig. 1, and let* $\mathcal{C}_1 = \Gamma^*$, $\mathcal{C}_2 = \{Z\}$. *Then we obtain the following system of equations (since* Δ *has only one control state* p, *we write* $\langle X, \bullet\rangle$ *and* $\langle X, \varepsilon\rangle$ *instead of* $\langle pX\bullet\rangle$ *and* $\langle pXp\rangle$, *resp.):*

$$\begin{aligned}
\langle Z,\bullet\rangle &= 1\\
\langle Z,\varepsilon\rangle &= x\langle I,\varepsilon\rangle\langle Z,\varepsilon\rangle + (1{-}x)\langle D,\varepsilon\rangle\langle Z,\varepsilon\rangle\\
\langle I,\bullet\rangle &= x(\langle I,\bullet\rangle + \langle I,\varepsilon\rangle\langle I,\bullet\rangle)\\
\langle I,\varepsilon\rangle &= x\langle I,\varepsilon\rangle\langle I,\varepsilon\rangle + 1{-}x\\
\langle D,\bullet\rangle &= (1{-}x)(\langle D,\bullet\rangle + \langle D,\varepsilon\rangle\langle D,\bullet\rangle)\\
\langle D,\varepsilon\rangle &= (1{-}x)\langle D,\varepsilon\rangle\langle D,\varepsilon\rangle + x
\end{aligned}$$

As the least solution we obtain the probabilities $[Z,\bullet] = 1$, $[Z,\varepsilon] = 0$, $[I,\bullet] = 0$, $[I,\varepsilon] = \min\{1,(1{-}x)/x\}$, $[D,\bullet] = 0$, $[D,\varepsilon] = \min\{1,x/(1{-}x)\}$. *By applying Lemma 3.3 we further obtain that, e.g.,* $\mathcal{P}(IIZ, \mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2) = \min\{1,(1{-}x)^2/x^2\}$.

In Example 3.5, the least solution of the constructed system of equations could be computed explicitly. This is generally impossible, but certain properties of the least solution are still decidable. For our purposes, it suffices to consider the class of properties defined in the next theorem.

Theorem 3.6. *Let* $Const = \mathbb{Q} \cup \{[pXq], [pX\bullet] \mid p, q \in Q \text{ and } X \in \Gamma\}$, *where* $\mathbb{Q}$ *is the set of all rational constants. Let* E_1, E_2 *be expressions built over* $Const$ *using '*$\cdot$*' and '*$+$*', and let* $\sim \in \{<, =\}$. *It is decidable whether* $E_1 \sim E_2$.

Proof. We show that, due to Theorem 3.4, $E_1 \sim E_2$ is effectively expressible as a closed formula of $(\mathbb{R}, +, *, \leq)$. Hence, the theorem follows from the decidability of first-order arithmetic of reals [25].

For all $p, q \in Q$ and $X \in \Gamma$, let $x(pXq)$, $x(pX\bullet)$, $y(pXq)$, and $y(pX\bullet)$ be first order variables, and let $\vec{X}$

and $\vec{Y}$ be the vectors of all $x(pXq)$, $x(pX\bullet)$, and $y(pXq)$, $y(pX\bullet)$ variables, respectively. Let us consider the formula Φ constructed as follows:

$$\exists \vec{X} \,:\, \vec{0} \leq \vec{X} \leq \vec{1} \quad \wedge \quad \vec{X} = \mathcal{F}(\vec{X})$$
$$\wedge \ (\forall \vec{Y} \,:\, (\vec{0} \leq \vec{Y} \leq \vec{1} \wedge \vec{Y} = \mathcal{F}(\vec{Y})) \Rightarrow \vec{X} \leq \vec{Y}))$$
$$\wedge \ E_1[\vec{X}/\pi] \sim E_2[\vec{X}/\pi]$$

Observe that the conditions $\vec{X} = \mathcal{F}(\vec{X})$ and $\vec{Y} = \mathcal{F}(\vec{Y})$ are expressible only using multiplication, summation, and equality. The expressions $E_1[\vec{X}/\pi]$ and $E_2[\vec{X}/\pi]$ are obtained from E_1 and E_2 by substituting all $[pXq]$ and $[pX\bullet]$ with $x(pXq)$ and $x(pX\bullet)$, respectively. It follows immediately that $E_1 \sim E_2$ iff Φ holds. □

Input: $pX \in \mathcal{C}(\Delta)$, $0 < \lambda < 1$
Output: $\mathcal{P}^\ell$, $\mathcal{P}^u$

```
1: P^ℓ := 0; P^u := 1;
2: for i = 1 to ⌈−log₂ λ⌉
3:     if [pX•] + Σ_{qε∈C₂} [pXq] ≥ (P^u − P^ℓ)/2
4:         then P^ℓ := (P^u − P^ℓ)/2
5:         else P^u := (P^u − P^ℓ)/2
6:     fi
```

Figure 2. Computing $\mathcal{P}^\ell, \mathcal{P}^u$

An immediate consequence of Theorem 3.6 is the following:

Theorem 3.7. *Let $p\alpha \in \mathcal{C}(\Delta)$, $\varrho \in [0,1]$, $\sim \in \{\leq, <, \geq, >\}$ and $0 < \lambda < 1$. It is decidable whether $\mathcal{P}(p\alpha, \mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2) \sim \varrho$. Moreover, there effectively exist rational numbers $\mathcal{P}^\ell, \mathcal{P}^u$ such that $\mathcal{P}^\ell \leq \mathcal{P}(p\alpha, \mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2) \leq \mathcal{P}^u$ and $\mathcal{P}^u - \mathcal{P}^\ell \leq \lambda$.*

Proof. We can assume w.l.o.g. that $\alpha = X$ for some $X \in \Gamma$. Note that $\mathcal{P}(pX, \mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2) \sim \varrho$ iff $[pX\bullet] + \sum_{q\varepsilon \in \mathcal{C}_2} [pXq] \sim \varrho$ by Lemma 3.3. Hence, we can apply Theorem 3.6. The numbers $\mathcal{P}^\ell$, $\mathcal{P}^u$ are computable, e.g., by the algorithm of Fig. 2. □

4. Model Checking PCTL for pPDA Processes

4.1. Qualitative Fragment of PCTL

For the rest of this section we fix a pPDA $\Delta = (Q, \Gamma, \delta, Prob)$.

Lemma 4.1. *Let $\mathcal{C} \subseteq \mathcal{C}(\Delta)$ be a simple set. The sets $\{p\alpha \in \mathcal{C}(\Delta) \mid \mathcal{P}(p\alpha, \mathcal{X}\mathcal{C}) = 1\}$ and $\{p\alpha \in \mathcal{C}(\Delta) \mid \mathcal{P}(p\alpha, \mathcal{X}\mathcal{C}) = 0\}$ are effectively regular.*

Proof. Immediate. □

Lemma 4.2. *Let $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathcal{C}(\Delta)$ be simple sets. The set $\{p\alpha \in \mathcal{C}(\Delta) \mid \mathcal{P}(p\alpha, \mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2) = 1\}$ is effectively regular.*

Proof. Let $R(pX) = \{q \in Q \mid [pXq] > 0\}$ for all $p \in Q$, $X \in \Gamma$. For each $i \in \mathbb{N}_0$ we define the set $S_i \subseteq \mathcal{C}(\Delta)$ inductively as follows:

- $S_0 = \{q\varepsilon \mid q\varepsilon \in \mathcal{C}_2\} \cup \{qX\alpha \mid [pX\bullet] = 1, \alpha \in \Gamma^*\}$
- $S_{i+1} = \{pX\beta \mid [pX\bullet] + \sum_{q \in R(pX)} [pXq] = 1$ and $\forall q \in R(pX) : q\beta \in S_i\}$

Using Lemma 3.3, we can easily check that $\bigcup_{i=0}^{\infty} S_i = \{p\alpha \in \mathcal{C}(\Delta) \mid \mathcal{P}(p\alpha, \mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2) = 1\}$. To see that the set $\bigcup_{i=0}^{\infty} S_i$ is effectively regular, for each $p \in Q$ we construct a finite automaton $\mathcal{M}_p$ such that $L(\mathcal{M}_p) = \{\alpha \in \Gamma^* \mid p\alpha \in \bigcup_{i=0}^{\infty} S_i\}$. A Δ-automaton $\mathcal{A}$ recognizing the set $\bigcup_{i=0}^{\infty} S_i$ can then be constructed using standard algorithms of automata theory (in particular, note that regular languages are effectively closed under reverse). The states of $\mathcal{M}_p$ are all subsets of Q, $\{p\}$ is the initial state, Γ is the input alphabet, final states are those $T \subseteq Q$ where for every $q \in T$ we have that $q\varepsilon \in \mathcal{C}_2$ (in particular, note that $\emptyset$ is a final state), and transition function is given by $T \xrightarrow{X} U$ iff for every $q \in T$ we have that $[qX\bullet] + \sum_{r \in R(qX)} [qXr] = 1$ and $U = \bigcup_{q \in T} R(qX)$. Note that $\emptyset \xrightarrow{X} \emptyset$ for each $X \in \Gamma$. The definition of $\mathcal{M}_p$ is effective due to Theorem 3.6. It is straightforward to check that $L(\mathcal{M}_p) = \{\alpha \in \Gamma^* \mid p\alpha \in \bigcup_{i=0}^{\infty} S_i\}$. □

Lemma 4.3. *Let $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathcal{C}(\Delta)$ be simple sets. The set $\{p\alpha \in \mathcal{C}(\Delta) \mid \mathcal{P}(p\alpha, \mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2) = 0\}$ is effectively regular.*

Proof. Let $R(pX) = \{q \in Q \mid [pXq] > 0\}$ for all $p \in Q$, $X \in \Gamma$. For each $i \in \mathbb{N}_0$ we define the set $S_i \subseteq \mathcal{C}(\Delta)$ inductively as follows:

- $S_0 = \{q\varepsilon \mid q\varepsilon \notin \mathcal{C}_2\}$
- $S_{i+1} = \{pX\beta \mid [pX\bullet] = 0$ and $\forall q \in R(pX) : q\beta \in S_i\}$

The fact $\bigcup_{i=0}^{\infty} S_i = \{p\alpha \in \mathcal{C}(\Delta) \mid \mathcal{P}(p\alpha, \mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2) = 0\}$ follows immediately from Lemma 3.3. The set $\bigcup_{i=0}^{\infty} S_i$ is effectively regular, which can be shown by constructing a finite automaton $\mathcal{M}_p$ recognizing the set $\{\alpha \in \Gamma^* \mid p\alpha \in \bigcup_{i=0}^{\infty} S_i\}$. This construction and the rest of the argument are very similar to the ones of the proof of Lemma 4.2. Therefore, they are not given explicitly. □

Theorem 4.4. *Let φ be a qualitative pCTL formula and ν a regular valuation. The set $\{p\alpha \in \mathcal{C}(\Delta) \mid p\alpha \models^\nu \varphi\}$ is effectively regular.*

Proof. By induction on the structure of φ. The cases when $\varphi \equiv \mathtt{tt}$ and $\varphi \equiv a$ follow immediately. For Boolean connectives we use the fact that regular sets are closed under complement and intersection. The other cases are

covered by Lemma 4.1, 4.2, and 4.3 (here we also need Lemma 3.2). □

4.2. Model Checking PCTL for pBPA Processes

In this section we provide an error tolerant model checking algorithm for PCTL formulae and pBPA processes. Since it is not so obvious what is meant by error tolerance in the context of PCTL model checking, this notion is defined formally.

Let $\mathcal{T} = (S, \to, Prob)$ be a probabilistic transition system and $0 < \lambda < 1$. For every negation-free PCTL formula φ and valuation ν we define the denotation of φ over $\mathcal{T}$ w.r.t. ν with *error tolerance* λ, denoted $[\![\varphi]\!]^\nu_\lambda$, in the same way as $[\![\varphi]\!]^\nu$. The only exception is $\varphi_1 \,\mathcal{U}^{\sim\varrho}\, \varphi_2$ where

- if $\sim \in \{<, \leq\}$, then

$$[\![\varphi_1 \,\mathcal{U}^{\sim\varrho}\, \varphi_2]\!]^\nu_\lambda = \{s \in S \mid \mathcal{P}(Run(s, [\![\varphi_1]\!]^\nu_\lambda \,\mathcal{U}\, [\![\varphi_2]\!]^\nu_\lambda)) \sim \varrho + \lambda\}$$

- if $\sim \in \{>, \geq\}$, then

$$[\![\varphi_1 \,\mathcal{U}^{\sim\varrho}\, \varphi_2]\!]^\nu_\lambda = \{s \in S \mid \mathcal{P}(Run(s, [\![\varphi_1]\!]^\nu_\lambda \,\mathcal{U}\, [\![\varphi_2]\!]^\nu_\lambda)) \sim \varrho - \lambda\}$$

Note that for every negation-free formula φ we have that $[\![\varphi]\!]^\nu \subseteq [\![\varphi]\!]^\nu_\lambda$. Negations can be "pushed inside" to atomic propositions using dual connectives (note that, e.g., $\neg(\varphi \,\mathcal{U}^{\geq\varrho}\, \psi)$ is equivalent to $\varphi \,\mathcal{U}^{<\varrho}\, \psi$), and for regular valuations we can further replace every $\neg a$ with a fresh proposition b where $\nu(b)$ is the complement of $\nu(a)$. Hence, we can assume w.l.o.g. that φ is negation-free.

An *error tolerant PCTL model checking algorithm* is an algorithm which, for each PCTL formula φ, valuation ν, $s \in S$, and $0 < \lambda < 1$, outputs YES/NO so that

- if $s \in [\![\varphi]\!]^\nu$, then the answer is YES;
- if the answer is YES, then $s \in [\![\varphi]\!]^\nu_\lambda$.

For the rest of this section, let us fix a pBPA $\Delta = (\Gamma, \delta, Prob)$. Since Δ has just one (or "none") control state p, we write $[X, \bullet]$ and $[X, \varepsilon]$ instead of $[pX\bullet]$ and $[pXp]$, respectively.

Lemma 4.5. *Let $\mathcal{C} \subseteq \mathcal{C}(\Delta)$ be a simple set, $\varrho \in [0,1]$, and $\sim \in \{\leq, <, \geq, >\}$. The set $\{\alpha \in \mathcal{C}(\Delta) \mid \mathcal{P}(\alpha, \mathcal{X}\mathcal{C}) \sim \varrho\}$ is effectively regular.*

Proof. Immediate. □

The following lemma presents the crucial part of the algorithm. This is the place where we need the assumption that Δ is stateless.

Lemma 4.6. *Let $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathcal{C}(\Delta)$ be simple sets. For all $\varrho \in [0,1]$ and $0 < \lambda < 1$ there effectively exist Δ-automata $\mathcal{A}^{\geq}$ and $\mathcal{A}^{\leq}$ such that for all $\alpha \in \mathcal{C}(\Delta)$ we have that*

- *if $\mathcal{P}(\alpha, \mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2) \geq \varrho$ (or $\mathcal{P}(\alpha, \mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2) \leq \varrho$), then $\alpha \in \mathcal{C}(\mathcal{A}^{\geq})$ (or $\alpha \in \mathcal{C}(\mathcal{A}^{\leq})$, respectively.)*
- *if $\alpha \in \mathcal{C}(\mathcal{A}^{\geq})$ (or $\alpha \in \mathcal{C}(\mathcal{A}^{\leq})$), then $\mathcal{P}(\alpha, \mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2) \geq \varrho - \lambda$ (or $\mathcal{P}(\alpha, \mathcal{C}_1 \,\mathcal{U}\, \mathcal{C}_2) \leq \varrho + \lambda$, respectively.)*

Theorem 4.7. *There is an error-tolerant PCTL model checking algorithm for pBPA processes.*

Proof. The proof is similar to the one of Theorem 4.4, using Lemma 4.5 and 4.6 instead of Lemma 4.1, 4.2, and 4.3. Note that Lemma 3.2 is applicable also to pBPA (the system Δ' constructed in Lemma 3.2 has the same set of control states as the original system Δ). □

5. Model Checking Deterministic Büchi Automata Specifications

Definition 5.1. *A* deterministic Büchi automaton *is a tuple $\mathcal{B} = (\Sigma, B, \varrho, b_I, Acc)$, where Σ is a finite* alphabet, *B is a finite set of* states, *$\varrho\colon B \times \Sigma \to B$ is a (total)* transition function *(we write $b \xrightarrow{a} b'$ instead of $\varrho(b,a) = b'$), b_I is the* initial state, *and $Acc \subseteq B$ is a set of* accepting states.

A run *of $\mathcal{B}$ is an infinite sequence $b_0 b_1 \ldots$ of states such that for every $i \geq 0$ there is $a \in \Sigma$ such that $b_i \xrightarrow{a} b_{i+1}$. A run $b_0 b_1 \ldots$ is* accepting *if $b_i \in Acc$ for infinitely many indices $i \geq 0$.*

For the rest of this section, we fix a pPDA $\Delta = (Q, \Gamma, \delta, Prob)$.

Definition 5.2. *Given a configuration $pX\alpha$ of Δ, we call pX the* head *and α the* tail *of $pX\alpha$. The set $Q \times \Gamma$ of all heads of Δ is also denoted by $\mathcal{H}(\Delta)$.*

We consider specifications given by deterministic Büchi automata having $\mathcal{H}(\Delta)$ as their alphabet. It is well known that every LTL formula whose atomic propositions are interpreted over simple sets can be encoded into a *nondeterministic* Büchi automaton having $\mathcal{H}(\Delta)$ as alphabet. Deterministic Büchi automata can encode the fragment of LTL that can also be expressed in the alternation-free modal μ-calculus [21]. Our results can be extended to atomic propositions interpreted over arbitrary regular sets of configurations using the same technique as in [15].

Definition 5.3. *The* product *of Δ and $\mathcal{B}$ is a probabilistic pushdown automaton $\Delta\mathcal{B} = (Q \times B, \Gamma, \delta', Prob')$ where δ' and $Prob'$ are determined as follows: $[p,b]X \xrightarrow{x} [p',b']\alpha$ iff $pX \xrightarrow{x} p'\alpha$ and $b \xrightarrow{pX} b'$ are transitions of Δ and $\mathcal{B}$, respectively.*

Notice that every (finite or infinite) path in $\mathcal{T}_{\Delta\mathcal{B}}$ corresponds to a unique path in $\mathcal{T}_\Delta$ obtained by projecting the control state of every configuration $[p,b]\alpha$ of the path onto its first component, yielding the configuration $p\alpha$. Conversely, for each path in $\mathcal{T}_\Delta$ (starting in some $p\alpha$) and each $b \in B$ there is exactly one path in $\mathcal{T}_{\Delta\mathcal{B}}$ starting in $[p,b]\alpha$ because $\mathcal{B}$ is deterministic.

Definition 5.4. *A configuration $[p,b]\alpha$ of $\Delta\mathcal{B}$ is* accepting *if $b \in Acc$. A run in $\mathcal{T}_{\Delta\mathcal{B}}$ is* accepting *if it visits accepting configurations infinitely often. A run in $\mathcal{T}_\Delta$ is accepting if its corresponding run in $\mathcal{T}_{\Delta\mathcal{B}}$ is accepting.*

The probability $\mathcal{P}(p\alpha,\mathcal{B})$ that a configuration $p\alpha$ of Δ satisfies the specification $\mathcal{B}$ is defined as $\mathcal{P}(p\alpha,\mathcal{B}) = \mathcal{P}(\{w \in Run(p\alpha) \mid w \text{ is accepting}\})$.

We solve the following two problems for a given configuration $p\alpha$ of Δ:

(a) Given $\varrho \in [0,1]$ and $\sim \in \{\leq,<,\geq,>,=\}$, do we have $\mathcal{P}(p\alpha,\mathcal{B}) \sim \varrho$?

(b) Given $0 < \lambda < 1$, compute rationals $\mathcal{P}^\ell, \mathcal{P}^u$ such that $\mathcal{P}^\ell \leq \mathcal{P}(p\alpha,\mathcal{B}) \leq \mathcal{P}^u$ and $\mathcal{P}^u - \mathcal{P}^\ell \leq \lambda$.

For finite-state automata, the problem can be solved as follows (see [11]). Let $\mathcal{A}$ be a finite-state automaton. Since the product automaton $\mathcal{A}\times\mathcal{B}$ is finite, it can be transformed into a finite Markov chain M by just 'copying' the probabilities of the system [11]. It is then possible to reduce problems (a),(b) to the problem of computing the probability of hitting a bottom strongly connected component of M which contains a state of the form (s,b), where b is accepting.

In our case, the product automaton $\Delta\mathcal{B}$ is again a pPDA, and so its associated probabilistic transition system is infinite. The key to our solution for (a) and (b) is the construction of a new finite Markov chain $M_{\Delta\mathcal{B}}$ that plays the rôle of M in the case of finite automata.

5.1. The Markov chain M_Δ

A *Büchi* pPDA is a tuple $\Delta = (Q,\Gamma,\delta,Prob,Acc_\Delta)$, where all elements except for Acc_Δ are defined as for pPDA and $Acc_\Delta \subseteq Q$ is a set of *accepting states*.

A configuration $p\alpha$ of Δ is *accepting* if $p \in Acc_\Delta$. A run of Δ is *accepting* if it visits accepting configurations infinitely often. For all $p \in Q$ and $X \in \Gamma$, the probability that a run $w \in Run(pX)$ is accepting is denoted by $\mathcal{P}(pX,Acc)$.

Obviously, the model checking problems (a),(b) of the previous section can be reduced to the following problems about a given configuration pX of a Büchi pPDA Δ (where $pX \in \mathcal{H}(\Delta)$):

(A) Given $\varrho \in [0,1]$ and $\sim \in \{\leq,<,\geq,>,=\}$, do we have $\mathcal{P}(pX,Acc) \sim \varrho$?

(B) Given $0 < \lambda < 1$, compute rationals $\mathcal{P}^\ell, \mathcal{P}^u$ such that $\mathcal{P}^\ell \leq \mathcal{P}(pX,Acc) \leq \mathcal{P}^u$ and $\mathcal{P}^u - \mathcal{P}^\ell \leq \lambda$.

For the rest of this section, we fix a Büchi pPDA $\Delta = (Q,\Gamma,\delta,Prob,Acc_\Delta)$.

Definition 5.5. *Let $w = p_0\alpha_0, p_1\alpha_1, \cdots$ be an infinite run in $\mathcal{T}_\Delta$. For each $i \in \mathbb{N}$ we define the i^{th}* minimum *of w, denoted $\min_i(w)$, inductively as follows:*

- $\min_1(w) = p_k\alpha_k$, *where $k \in \mathbb{N}_0$ is the least number such that $|\alpha_{k'}| \geq |\alpha_k|$ for each $k' \geq k$.*
- $\min_{i+1}(w) = \min_1(w_{\ell+1})$, *where $\min_i(w) = p_\ell\alpha_\ell$. Here $w_{\ell+1}$ is the suffix of w that starts with $p_{\ell+1}\alpha_{\ell+1}$.*

We say that w flashes *at $\min_i(w)$ if either $i = 1$ and $\min_1(w)$ is accepting, or $i > 1$ and w visits an accepting configuration between $\min_{i-1}(w)$ and $\min_i(w)$ (where $\min_{i-1}(w)$ is not included).*

Sometimes we abuse language and use $\min_i(w)$ to denote not only a configuration, but the particular *occurrence* of the configuration that corresponds to the i^{th} minimum.

For all $pX \in \mathcal{H}(\Delta)$ and all $i \in \mathbb{N}$ we define a random variable $V_{pX}^{(i)}$ over $Run(pX)$ as follows: The possible values of $V_{pX}^{(i)}$ are all pairs of the form (qY,f), where $qY \in \mathcal{H}(\Delta)$ and $f \in \{0,1\}$ is a boolean flag; there is also a special value $\perp$. For a given $w \in Run(pX)$, $V_{pX}^{(i)}(w)$ is determined as follows:

- if w is finite then $V_{pX}^{(i)}(w) = \perp$;
- if conditions (1)–(3) below are satisfied, then $V_{pX}^{(i)}(w) = (qY,1)$;

  (1) w is infinite;
  (2) the head of $\min_i(w)$ is qY;
  (3) w flashes at $\min_i(w)$.

- if conditions (1) and (2) above are satisfied and condition (3) is not satisfied, then $V_{pX}^{(i)}(w) = (qY,0)$

Notice that the random variables are well defined, because they assign to each run exactly one value.

Lemma 5.6. *For all $pX \in \mathcal{H}(\Delta)$, $n \in \mathbb{N}$, and $v_1,\cdots,v_n$, the probability of $V_{pX}^{(1)}{=}v_1 \wedge \cdots \wedge V_{pX}^{(n)}{=}v_n$ exists (i.e., the set of all $w \in Run(pX)$ which satisfy this condition is $\mathcal{P}$-measurable). Moreover, for every rational constant y there is an effectively constructible formula of $(\mathbb{R},+,*,\leq)$ which holds if and only if $\mathcal{P}(V_{pX}^{(1)}{=}v_1 \wedge \cdots \wedge V_{pX}^{(n)}{=}v_n) = y$.*

The following lemma proves the Markov property. In fact, it follows immediatelly from the construction used in the proof of Lemma 5.6.

Lemma 5.7. *The conditional probability of $V_{pX}^{(n)} = v_n$ on the hypothesis $V_{pX}^{(1)} = v_1 \wedge \cdots \wedge V_{pX}^{(n-1)} = v_{n-1}$ is equal to the probability of $V_{pX}^{(n)} = v_n$ conditioned on $V_{pX}^{(n-1)} = v_{n-1}$, assuming that the probability of $V_{pX}^{(1)} = v_1 \wedge \cdots \wedge V_{pX}^{(n-1)} = v_{n-1}$ is non-zero.*

For each control state $q \in Q$ we define a flag f_q, which is equal either to 1 or 0 depending on whether $q \in Acc_\Delta$ or not, respectively. Another consequence of Lemma 5.6 is the following:

Lemma 5.8. *The conditional probability of* $V_{pX}^{(n)}{=}(q'Y', f')$ *on the hypothesis* $V_{pX}^{(n-1)}{=}(qY, f))$ *is equal to the conditional probability of* $V_{qY}^{(2)}{=}(q'Y', f')$ *on the hypothesis* $V_{qY}^{(1)}{=}(qY, f_q)$*, assuming that* $\mathcal{P}(V_{pX}^{(n-1)}{=}(qY, f)) \neq 0$.

Now we can define the finite Markov chain M_Δ.

Definition 5.9. *Let* M_Δ *be a finite-state Markov chain where the set of states is*

$$\begin{array}{ll} & \{(qY, 0) \mid q \notin Acc_\Delta, Y \in \Gamma, \mathcal{P}(V_{qY}^{(1)}{=}(qY, 0)) > 0\} \\ \cup & \{(qY, 1) \mid qY \in \mathcal{H}(\Delta), \mathcal{P}(V_{qY}^{(1)}{=}(qY, f_q)) > 0\} \\ \cup & \mathcal{H}(\Delta) \cup \{\bot\} \end{array}$$

and transition probabilities are defined as follows:

- $Prob(\bot \to \bot) = 1$,
- $Prob(pX \to (qY, f)) = \mathcal{P}(V_{pX}^{(1)}{=}(qY, f))$,
- $Prob(pX \to \bot) = \mathcal{P}(V_{pX}^{(1)}{=}\bot)$,
- $Prob((qY, f) \to (q'Y', f')) = \mathcal{P}(V_{qY}^{(2)}{=}(q'Y', f') \mid V_{qY}^{(1)}{=}(qY, f_q))$.

One can readily check that M_Δ is indeed a Markov chain, i.e., for every state s of M_Δ we have that the sum of probabilities of all outgoing transitions of s is equal to one.

A *trajectory* in M_Δ is an infinite sequence $\sigma(0)\,\sigma(1)\,\cdots$ of states of M_Δ where $Prob(\sigma(i) \to \sigma(i+1)) > 0$ for every $i \in \mathbb{N}_0$.

To every run $w \in Run(pX)$ of Δ we associate its *footprint*, which is an infinite sequence σ of states of M_Δ defined as follows:

- $\sigma(0) = pX$
- if w is finite, then for every $i \in \mathbb{N}$ we have $\sigma(i) = \bot$;
- if w is infinite, then for every $i \in \mathbb{N}$ we have $\sigma(i) = (p_iX_i, f_i)$, where p_iX_i is the head of $\min_i(w)$, and $f_i = 1$ iff w flashes at $\min_i(w)$.

We say that a given $w \in Run(pX)$ is *good* if the footprint of w is a trajectory in M_Δ. Our next lemma reveals that almost all runs are good.

Lemma 5.10. *Let* $pX \in \mathcal{H}(\Delta)$*. We have that* $\mathcal{P}(\{w \in Run(pX) \mid w \text{ is good}\}) = 1$.

It follows directly from the definition of M_Δ that if both $(qY, 0)$ and $(qY, 1)$ are states of M_Δ, then they have the "same" outgoing arcs (i.e., $(qY, 0) \xrightarrow{x} (rZ, f)$ iff $(qY, 1) \xrightarrow{x} (rZ, f)$, where $x > 0$). In particular, this means that if $(qY, 0)$ or $(qY, 1)$ belongs to some strongly connected component C of M_Δ, then all of the outgoing arcs of $(qY, 0)$ and $(qY, 1)$ lead to C. Hence, the following definition is correct:

Definition 5.11. *We say that a given* $qY \in \mathcal{H}(\Delta)$ *is* recurrent *if there is a bottom strongly connected component* C_{qY} *of* M_Δ *such that* $(qY, f) \in C_{qY}$ *for some* $f \in \{0, 1\}$.

Each recurrent head is either accepting *or* rejecting*, depending on whether* C_{qY} *contains a state of the form* $(rZ, 1)$ *or not, respectively.*

We say that a run w of Δ *hits* a head $qY \in \mathcal{H}(\Delta)$ if there is some $i \in \mathbb{N}$ such that the head of $\min_i(w)$ is qY. The next lemma says that an infinite run eventually hits a recurrent head.

Lemma 5.12. *Let* $pX \in \mathcal{H}(\Delta)$*. The conditional probability that* $w \in Run(pX)$ *hits a recurrent head on the hypothesis that* w *is infinite is equal to one.*

So, an infinite run eventually hits a recurrent head. Now we prove that if this head is accepting/rejecting, then the run will be accepting/rejecting with probability one.

Lemma 5.13. *Let* qY *be an accepting/rejecting head. The conditional probability that* $w \in Run(pX)$ *is accepting/rejecting on the hypothesis that the first recurrent head hit by* w *is accepting/rejecting is equal to one.*

Lemma 5.14. *(cf. Proposition 4.1.5 of [11]) Let* $pX \in \mathcal{H}(\Delta)$*.* $\mathcal{P}(pX, Acc)$ *is equal to the probability that a trajectory from* pX *in* M_Δ *hits a state of the form* (qY, f) *where* qY *is an accepting head (this is equivalent to saying that the trajectory hits a bottom strongly connected component of* M_Δ *which contains a state of the form* $(rZ, 1)$*).*

Theorem 5.15. *Let* Δ *be a Büchi pPDA. Given a head* $pX \in \mathcal{H}(\Delta)$*,* $\sim \in \{\leq, <, \geq, >, =\}$*, and* $\varrho \in [0, 1]$*, we can decide if* $\mathcal{P}(pX, Acc) \sim \varrho$*. Further, for each* $0 < \lambda < 1$ *we can compute rationals* $\mathcal{P}^\ell, \mathcal{P}^u$ *such that* $\mathcal{P}^\ell \leq \mathcal{P}(pX, Acc) \leq \mathcal{P}^u$ *and* $\mathcal{P}^u - \mathcal{P}^\ell \leq \lambda$*.*

Proof. Similarly as in Theorem 3.6, we compute a closed formula Φ of $(\mathbb{R}, +, *, \leq)$ such that $\mathcal{P}(pX, Acc) \sim \varrho$ iff Φ holds. Then, the rationals $\mathcal{P}^\ell, \mathcal{P}^u$ can be computed by a simple binary search similarly as in Fig. 2.

Due to Lemma 5.14 we know that $\mathcal{P}(pX, Acc) = \mathcal{P}(pX, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2)$, where $\mathcal{C}_1$ is the set of all states of M_Δ, and $\mathcal{C}_2$ consists of all states of the form (qY, f) where qY is an accepting head. This means that there is a system of recursive equations such that $\mathcal{P}(pX, Acc)$ appears in the tuple of values which form the least solution of the system (we can assume that $\mathcal{P}(pX, Acc)$ is, e.g., the first element of this tuple). Since M_Δ is finite, these equations are linear and by using the results of [17, 11] we can even assume that there is a unique solution. The only problem is that numeric coefficients in this system of equations are the probabilities of transitions in M_Δ which cannot be precisely computed. This can be overcome as follows: we construct the mentioned system of linear equations where we replace each coefficient with a fresh first-order variable; let $\vec{C}$ be the tuple

of all variables which correspond to the coefficients. Now we can effectively construct the formula

$$\Psi \quad \equiv \quad \exists \vec{Z} : \vec{Z} = \mathcal{L}(\vec{Z}) \ \wedge \ \vec{Z}_1 \sim \varrho$$

where $\vec{Z} = \mathcal{L}(\vec{Z})$ says that the tuple of variables $\vec{Z}$ is a solution of the constructed system of linear equations. Note that Ψ is not closed because the variables of $\vec{C}$ (which appear in the $\vec{Z} = \mathcal{L}(\vec{Z})$ subformula) are free. Due to Lemma 5.6, for each of these coefficients there effectively exists a formula of $(\mathbb{R}, +, *, \leq)$ which says that a given coefficient is equal to the probability of the corresponding transition in M_Δ (we just "translate" the definition of $Prob$ given in Definition 5.9 into $(\mathbb{R}, +, *, \leq)$, using the formulae provided by Lemma 5.6). Let $\Psi_{\vec{C}}$ by a conjunction of all these formulae. The formula Φ is constructed as follows:

$$\Phi \quad \equiv \quad \exists \vec{Z} : \exists \vec{C} : \Psi_{\vec{C}} \ \wedge \ \vec{Z} = \mathcal{L}(\vec{Z}) \ \wedge \ \vec{Z}_1 \sim \varrho$$

Obviously, $\mathcal{P}(pX, Acc) \sim \varrho$ iff Φ holds. □

We conclude this section by trying to explain why our results cannot be directly extended to nondeterministic Büchi automata. First of all, notice that we cannot assign probabilities to the transitions of $\Delta\mathcal{B}$ in a meaningful way, because a transition $p\alpha \xrightarrow{x} q\beta$ of Δ should 'split' into several transitions of $\Delta\mathcal{B}$. In the case of a finite automaton $\mathcal{A}$, this problem can be solved by working with the product of $\mathcal{A}$ and $d\mathcal{B}$, where $d\mathcal{B}$ is the result of applying the determinization construction to $\mathcal{B}$. Let $\mathcal{A}d\mathcal{B}$ denote this product. In [11], a definition of recurrence is provided, which characterizes the states $[s, b]$ of $\mathcal{A}d\mathcal{B}$ that, loosely speaking, return to $[s, b]$ with probability 1 in terms of topological properties of the probabilistic transition system $\mathcal{A}d\mathcal{B}$. It is then possible to compute the accepting recurrent states.

Unfortunately, this construction does not seem to generalize to the case of pushdown automata. The problem is that the Büchi pPDA $\Delta\, d\mathcal{B}$ has infinitely many states, and so it must be replaced by the chain $M_{\Delta\, d\mathcal{B}}$. However, in $M_{\Delta\, d\mathcal{B}}$ we cannot directly 'observe' the points at which a run hits an accepting state; we can only observe the points at which a run hits a minimum. While we can use $M_{\Delta\, d\mathcal{B}}$ to compute the recurrent minima, i.e., the heads to which one can return with probability 1 at a minimum, at the moment we do not know how to compute the accepting recurrent minima, i.e., the recurrent minima that not only return, but also visit an accepting configuration along the way. More precisely, we know how to decide for a given head pX if the runs starting at it will almost surely hit *some* head p_iX_i out of a set $\mathcal{H} = \{p_1X_1, \ldots, p_nX_n\}$ and visit some accepting configuration along the way. We can also decide if *some* head $p_jX_j \in \mathcal{H}(\Delta)$ will hit pX with probability one. However, since we do not know whether $i = j$ or not, this information is not sufficient to decide if pX is an accepting recurrent minimum.

6. Conclusions

We have provided model checking algorithms for pushdown automata against PCTL specifications, and against linear-time specifications represented as deterministic Büchi automata. Contrary to the case of finite automata, qualitative properties (i.e., whether a property holds with probability 0 or 1), depend on the exact probabilities of the transitions.

There are many possibilities for future work. An obvious question is what is the complexity of the obtained algorithms. Since the formulae of first order arithmetic which are constructed in our algorithms have a fixed alternation depth, we can apply a powerful result of Grigoriev [16] which says that the validity of such formulae is decidable in single exponential time. From this we can easily derive the time complexity of some of our algorithms (for example, the qualitative/quantitative random walk problem of Section 3 is decidable in exponential time). Since the complexity issues were not the main priority of our work, the efficiency of our algorithms can be improved even by relatively straightforward optimizations. Moreover, there is a lot of space for advanced numerical algorithms which might be used to compute the required probabilities with enough precision.

An obvious question about linear-time specifications is whether our procedure can be improved to deal with nondeterministic Büchi automata. Another possibility is to consider LTL specifications and try to generalize the technique of [11], which modifies the probabilistic transition systems step-by-step and at the same time simplifies the formula, until it becomes a propositional formula.

7. Acknowledgments

The authors would like to thank Stefan Schwoon for many helpful insights.

References

[1] P.A. Abdulla, C. Baier, S.P. Iyer, and B. Jonsson. Reasoning about probabilistic channel systems. In *Proceedings of CONCUR 2000*, vol. 1877 of *Lecture Notes in Computer Science*, pp. 320–330. Springer, 2000.

[2] P.A. Abdulla and A. Rabinovich. Verifi cation of probabilistic systems with faulty communication. In *Proceedings of FoSSaCS 2003*, vol. 2620 of *Lecture Notes in Computer Science*, pp. 39–53. Springer, 2003.

[3] A. Abney, D. McAllester, and F. Pereira. Relating probabilistic grammars and automata. In *Proceedings of ACP'99*, pp. 542–549, 1999.

[4] R. Alur, K. Etessami, and M. Yannakakis. Analysis of recursive state machines. In *Proceedings of CAV 2001*, vol. 2102 of *Lecture Notes in Computer Science*, pp. 207–220. Springer, 2001.

[5] A. Aziz, V. Singhal, F. Balarin, R. Brayton, and A. Sangiovanni-Vincentelli. It usually works: The temporal logic of stochastic systems. In *Proceedings of CAV'95*, vol. 939 of *Lecture Notes in Computer Science*, pp. 155–165. Springer, 1995.

[6] J.C.M. Baeten and W.P. Weijland. *Process Algebra*. No. 18 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1990.

[7] C. Baier and B. Engelen. Establishing qualitative properties for probabilistic lossy channel systems: an algorithmic approach. In *Proceedings of 5th International AMAST Workshop on Real-Time and Probabilistic Systems (ARTS'99)*, vol. 1601 of *Lecture Notes in Computer Science*, pp. 34–52. Springer, 1999.

[8] N. Bertrand and Ph. Schnoebelen. Model checking lossy channel systems is probably decidable. In *Proceedings of FoSSaCS 2003*, vol. 2620 of *Lecture Notes in Computer Science*, pp. 120–135. Springer, 2003.

[9] O. Burkart and B. Steffen. Model checking the full modal mu-calculus for infinite sequential processes. In *Proceedings of ICALP'97*, vol. 1256 of *Lecture Notes in Computer Science*, pp. 419–429. Springer, 1997.

[10] C. Courcoubetis and M. Yannakakis. Verifying temporal properties of finite-state probabilistic programs. In *Proceedings of 29th Annual Symposium on Foundations of Computer Science*, pp. 338–345. IEEE Computer Society Press, 1988.

[11] C. Courcoubetis and M. Yannakakis. The complexity of probabilistic verification. *Journal of the Association for Computing Machinery*, 42(4):857–907, 1995.

[12] J.M. Couvreur, N. Saheb, and G. Sutre. An optimal automata approach to LTL model checking of probabilistic systems. In *Proceedings of LPAR 2003*, vol. 2850 of *Lecture Notes in Computer Science*, pp. 361–375. Springer, 2003.

[13] J. Esparza, D. Hansel, P. Rossmanith, and S. Schwoon. Efficient algorithms for model checking pushdown systems. In *Proceedings of CAV 2000*, vol. 1855 of *Lecture Notes in Computer Science*, pp. 232–247. Springer, 2000.

[14] J. Esparza, A. Kučera, and R. Mayr. Model checking probabilistic pushdown automata. Technical Report FIMU-RS-2004-03, Faculty of Informatics, Masaryk University, 2004.

[15] J. Esparza, A. Kučera, and S. Schwoon. Model-checking LTL with regular valuations for pushdown systems. *Information and Computation*, 186(2):355–376, 2003.

[16] D. Grigoriev. Complexity of deciding Tarski algebra. *Journal of Symbolic Computation*, 5(1–2):65–108, 1988.

[17] H. Hansson and B. Jonsson. A logic for reasoning about time and reliability. *Formal Aspects of Computing*, 6:512–535, 1994.

[18] S. Hart and M. Sharir. Probabilistic temporal logic for finite and bounded models. In *Proceedings of POPL'84*, pp. 1–13. ACM Press, 1984.

[19] M. Huth and M.Z. Kwiatkowska. Quantitative analysis and model checking. In *Proceedings of LICS'97*, pp. 111–122. IEEE Computer Society Press, 1997.

[20] S.P. Iyer and M. Narasimha. Probabilistic lossy channel systems. In *Proceedings of TAPSOFT'97*, vol. 1214 of *Lecture Notes in Computer Science*, pp. 667–681. Springer, 1997.

[21] O. Kupferman and M.Y. Vardi. Freedom, weakness, and determinism: From linear-time to branching-time. In *Proceedings of LICS'98*, pp. 81–92. IEEE Computer Society Press, 1998.

[22] D. Lehman and S. Shelah. Reasoning with time and chance. *Information and Control*, 53:165–198, 1982.

[23] I. Macarie and M. Ogihara. Properties of probabilistic pushdown automata. *Theoretical Computer Science*, 207:117–130, 1998.

[24] A. Rabinovich. Quantitative analysis of probabilistic lossy channel systems. In *Proceedings of ICALP 2003*, vol. 2719 of *Lecture Notes in Computer Science*, pp. 1008–1021. Springer, 2003.

[25] A. Tarski. *A Decision Method for Elementary Algebra and Geometry*. Univ. of California Press, Berkeley, 1951.

[26] M. Vardi. Automatic verification of probabilistic concurrent finite-state programs. In *Proceedings of 26th Annual Symposium on Foundations of Computer Science*, pp. 327–338. IEEE Computer Society Press, 1985.

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