



FI MU

Faculty of Informatics
Masaryk University

Model Checking Probabilistic Pushdown Automata

by

Javier Esparza
Antonín Kučera
Richard Mayr

Model Checking Probabilistic Pushdown Automata

Javier Esparza*

Antonín Kučera†

Richard Mayr‡

July 28, 2004

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.

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

†Faculty of Informatics, Masaryk University, Botanická 68a, CZ-60200 Brno, Czech Republic, tony@fi.muni.cz. 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.

‡Department of Computer Science, Albert-Ludwigs-University Freiburg Georges-Koehler-Allee 51, D-79110 Freiburg, Germany. mayrri@informatik.uni-freiburg.de. Supported by Landesstiftung Baden-Württemberg, grant No. 21-655.023.

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 [LS82, HS84, Var85, CY88, HJ94, ASB⁺95, CY95, HK97, CSS03]. As for infinite-state systems, most works so far considered probabilistic lossy channel systems [IN97] which model asynchronous communication through unreliable channels [BE99, ABIJ00, AR03, BS03]. A notable recent result is the decidability of quantitative model checking of liveness properties specified by Büchi-automata for probabilistic lossy channel systems [Rab03]. 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 [AEY01, BS97, EHRS00, Wal01]), but the probabilistic extension has so far not been considered. As a related work we can mention [MO98], 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 [AMP99] 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 pBPA¹) and various probabilistic logics. We start with a

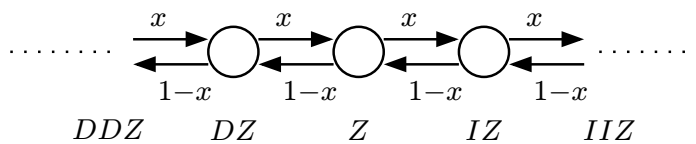


Figure 1: Bernoulli random walk as a pBPA

class of 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

¹This is a standard notation adopted in concurrency theory. The subclass of stateless PDA corresponds to a natural subclass of ACP known as Basic Process Algebra [BW90].

just three stack symbols Z, I, D and transitions $Z \xrightarrow{x} IZ, Z \xrightarrow{1-x} DZ, I \xrightarrow{x} II, I \xrightarrow{1-x} \varepsilon, D \xrightarrow{1-x} DD, \text{ and } D \xrightarrow{x} \varepsilon$, 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 [HJ94, CY95]. 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.

2 Preliminary Definitions

Definition 2.1. A probabilistic transition system is a triple $\mathcal{T} = (S, \rightarrow, Prob)$ where S is a finite or countably infinite set of states, $\rightarrow \subseteq S \times S$ is a transition relation, and $Prob$ is a function which to each transition $s \rightarrow t$ of \mathcal{T} assigns its probability $Prob(s \rightarrow t) \in (0, 1]$ so that for every $s \in S$ we have

$$\sum_{s \rightarrow t} Prob(s \rightarrow t) \in \{0, 1\}$$

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 $Prob(s \rightarrow t) = x$. A *path* in \mathcal{T} is a finite or infinite sequence $w = s_0, s_1, \dots$ of states such that $s_i \rightarrow s_{i+1}$ 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, all runs, and all infinite runs of \mathcal{T} are denoted $FPath$, Run , and $IRun$, respectively². Similarly, the sets of all finite paths, runs, and infinite runs that start in a given $s \in S$ are denoted $FPath(s)$, $Run(s)$, and $IRun(s)$, 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 $\mathcal{P} : \mathcal{F} \rightarrow [0, 1]$ is the unique probability function such that $\mathcal{P}(Run(w)) = \prod_{i=0}^{m-1} x_i$ where $w = s_0, \dots, s_m$ and $s_i \xrightarrow{x_i} s_{i+1}$ for every $0 \leq i < m$ (if $m = 0$, we put $\mathcal{P}(Run(w)) = 1$).

2.1 The Logic PCTL

PCTL, the probabilistic extension of CTL, was defined by Hansson & Jonsson in [HJ94]. Let $Ap = \{a, b, c, \dots\}$ be a countably infinite set of *atomic propositions*. The syntax of PCTL³ is given by the following abstract syntax equation:

$$\varphi ::= \text{tt} \mid a \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \mathcal{X}^{\varrho}\varphi \mid \varphi_1 \mathcal{U}^{\varrho}\varphi_2$$

Here a ranges over Ap , $\varrho \in [0, 1]$, and $\sim \in \{\leq, <, \geq, >\}$. Let $\mathcal{T} = (S, \rightarrow, Prob)$ 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

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

³For simplicity we omit the bounded ‘until’ operator of [HJ94].

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

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

$$\begin{aligned} \llbracket \mathbf{tt} \rrbracket^\nu &= S \\ \llbracket a \rrbracket^\nu &= \nu(a) \\ \llbracket \neg \varphi \rrbracket^\nu &= S \setminus \llbracket \varphi \rrbracket^\nu \\ \llbracket \varphi_1 \wedge \varphi_2 \rrbracket^\nu &= \llbracket \varphi_1 \rrbracket^\nu \cap \llbracket \varphi_2 \rrbracket^\nu \\ \llbracket \mathcal{X}^{\sim \varrho} \varphi \rrbracket^\nu &= \{s \in S \mid \mathcal{P}(Run(s, \mathcal{X} \llbracket \varphi \rrbracket^\nu)) \sim \varrho\} \\ \llbracket \varphi_1 \mathcal{U}^{\sim \varrho} \varphi_2 \rrbracket^\nu &= \{s \in S \mid \mathcal{P}(Run(s, \llbracket \varphi_1 \rrbracket^\nu \mathcal{U} \llbracket \varphi_2 \rrbracket^\nu)) \sim \varrho\} \end{aligned}$$

As usual, we write $s \models^\nu \varphi$ instead of $s \in \llbracket \varphi \rrbracket^\nu$.

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, $a \mathcal{U}^{<1} b \equiv \neg(a \mathcal{U}^{\geq 1} b)$.)

2.2 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 $pX \rightarrow q\alpha$ instead of $(p, X, q, \alpha) \in \delta$), and $Prob$ is a function which to each transition $pX \rightarrow q\alpha$ assigns its probability $Prob(pX \rightarrow q\alpha) \in (0, 1]$ so that for all $p \in Q$ and $X \in \Gamma$ we have that $\sum_{pX \rightarrow q\alpha} Prob(pX \rightarrow q\alpha) \in \{0, 1\}$.

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 $Prob(pX \rightarrow q\alpha) = x$. 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 $pX \rightarrow q\alpha \in \delta$, then $|\alpha| \leq 2$.

To Δ we associate the probabilistic transition system \mathcal{T}_Δ 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. $Q \subseteq St$, $\gamma : St \times \Gamma \rightarrow St$ 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. This is a standard result (see, e.g., [EKS03]). For the sake of completeness, we include an explicit proof.

Lemma 3.2. For each pPDA $\Delta = (Q, \Gamma, \delta, Prob)$ and regular sets $\mathcal{C}_1, \dots, \mathcal{C}_k \subseteq \mathcal{C}(\Delta)$ there effectively exist a pPDA $\Delta' = (Q, \Gamma', \delta', Prob')$, simple sets $\mathcal{C}'_1, \dots, \mathcal{C}'_k \subseteq \mathcal{C}(\Delta')$, and an injective mapping $\mathcal{G} : \mathcal{C}(\Delta) \rightarrow \mathcal{C}(\Delta')$ 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.

Proof. For each $1 \leq i \leq k$, let $\mathcal{A}_i = (St_i, \gamma_i, Acc_i)$ be a Δ -automaton such that $\mathcal{C}(\mathcal{A}_i) = \mathcal{C}_i$. Let $States = \prod_{i=1}^k \prod_{p \in Q} St_i$. For given $\vec{s} \in States$, $0 \leq i \leq k$, and $p \in Q$, we denote by $\vec{s}(i, p)$ the component of \vec{s} which corresponds to the i and p .

We put $\Gamma' = \Gamma \times States$. The transition function δ' and probabilities $Prob'$ are defined as follows:

- if $pX \xrightarrow{x} q\epsilon \in \delta$, then $p(X, \vec{s}) \xrightarrow{x} q\epsilon$ for each $\vec{s} \in States$;
- if $pX \xrightarrow{x} qY \in \delta$, then $p(X, \vec{s}) \xrightarrow{x} q(Y, \vec{s})$ for each $\vec{s} \in States$;
- if $pX \xrightarrow{x} qYZ \in \delta$, then $p(X, \vec{s}) \xrightarrow{x} q(Y, \vec{t})(Z, \vec{s})$ for all $\vec{s}, \vec{t} \in States$ such that $\gamma_i(\vec{s}(i, r), Z) = \vec{t}(i, r)$ for all $1 \leq i \leq k$ and $r \in Q$.

So, the Δ -automata $\mathcal{A}_1, \dots, \mathcal{A}_k$ are simulated “on-the-fly” by storing the vector of current states directly in the stack. Hence, the information whether a given \mathcal{A}_i accepts the current configuration is available in the topmost stack symbol. For every $0 \leq i \leq k$, the underlying set G_i of \mathcal{C}'_i (see Definition 3.1) is defined by

$$G_i = \{p(X, \vec{s}) \mid \gamma_i(\vec{s}(i, p), X) \in Acc_i\} \cup \{p\epsilon \mid p\epsilon \in \mathcal{C}_i\}$$

The function \mathcal{G} is defined by $\mathcal{G}(p\epsilon) = p\epsilon$, and $\mathcal{G}(pX_1 \dots X_k) = p(X_1, \vec{s}_1) \dots (X_k, \vec{s}_k)$, where $\vec{s}_k(i, q) = q$, and $\vec{s}_j(i, q) = \gamma_i(\vec{s}_{j+1}(i, q), X_{j+1})$ for all $1 \leq j < k$. It follows immediately from the definition δ' and $Prob'$ that the parts of \mathcal{T}_Δ and $\mathcal{T}_{\Delta'}$ which are reachable from $p\alpha$ and $\mathcal{G}(p\alpha)$ are isomorphic (for every $p\alpha \in \mathcal{C}(\Delta)$).

Let $\mathcal{C} \subseteq \mathcal{C}(\Delta')$ be a regular set of configurations. Since some configurations of \mathcal{C} can be “inconsistent” in the sense that the vectors of states that are stored together with the original stack symbols do not correspond to a valid computation of the \mathcal{A}_i automata, the set $\mathcal{G}^{-1}(\mathcal{C})$ is not a simple projection of \mathcal{C} “forgetting” the vectors of states from the stack symbols. Fortunately, $\mathcal{G}(\mathcal{C}(\Delta))$ is (obviously) a regular set, so we can construct a Δ' -automaton recognizing the set $\mathcal{C} \cap \mathcal{G}(\mathcal{C}(\Delta))$ and apply the mentioned projection. \square

For the rest of this section, let $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathcal{C}(\Delta)$ be (fixed) simple sets, and let $G_1, G_2 \subseteq Q \times (\Gamma \cup \{\epsilon\})$ be the sets associated to $\mathcal{C}_1, \mathcal{C}_2$ in the sense of Definition 3.1.

Definition 3.3. *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 every $\mathcal{C} \subseteq \mathcal{C}(\Delta)$ and every $\beta \in \Gamma^*$, the symbol $\mathcal{C}\beta$ denotes the set $\{p\alpha\beta \mid p\alpha \in \mathcal{C}\}$.
- 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 $\mathcal{P}(pX, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2^\bullet)$.
- Let A be a set of finite paths which end in the same state t , and let B a set of finite or infinite paths that start in t . Then the symbol $A \odot B$ denotes the set of paths $\{vw \mid v \in A, tw \in B\}$.

The following auxiliary lemmas are used in the proof of Lemma 3.6:

Lemma 3.4. *Let $\mathcal{T} = (S, \rightarrow, Prob)$ be a probabilistic transition system. Let $s, t \in S$ and $\mathcal{C}_1, \mathcal{C}_2 \subseteq S$. Further, let $A = FPath(s, (\mathcal{C}_1 \setminus \mathcal{C}_2) \mathcal{U} \{t\})$ and $B = FPath(t, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2)$. Then*

$$\sum_{w \in A \odot B} \mathcal{P}(Run(w)) = \sum_{w \in A} \mathcal{P}(Run(w)) \cdot \sum_{w \in B} \mathcal{P}(Run(w)).$$

Proof. Immediate. □

Lemma 3.5. *For all $p\alpha \in \mathcal{C}(\Delta)$ and $\beta \in \Gamma^*$ we have that $\mathcal{P}(Run(p\alpha, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2)) = \mathcal{P}(Run(p\alpha\beta, \mathcal{C}_1^\bullet \beta \mathcal{U} \mathcal{C}_2\beta))$.*

Proof. For every finite path $w = p_1\alpha_1 \cdots p_n\alpha_n$ of $FPath(p\alpha)$, let $w^{+\beta}$ denote the finite path $p_1\alpha_1\beta \cdots p_n\alpha_n\beta$ of $FPath(p\alpha\beta)$. Realize that $\mathcal{P}(Run(w)) = \mathcal{P}(Run(w^{+\beta}))$, because w and $w^{+\beta}$ execute the same transitions. One can easily verify that $w \in FPath(p\alpha, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2)$ iff $w^{+\beta} \in FPath(p\alpha\beta, \mathcal{C}_1^\bullet \beta \mathcal{U} \mathcal{C}_2\beta)$. From this we get $\mathcal{P}(Run(p\alpha, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2)) = \sum_{w \in FPath(p\alpha, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2)} \mathcal{P}(Run(w)) = \sum_{w \in FPath(p\alpha\beta, \mathcal{C}_1^\bullet \beta \mathcal{U} \mathcal{C}_2\beta)} \mathcal{P}(Run(w)) = \mathcal{P}(Run(p\alpha\beta, \mathcal{C}_1^\bullet \beta \mathcal{U} \mathcal{C}_2\beta))$. □

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 3.6. *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*

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

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

Proof. By induction on n . For $n = 0$ we have that $\mathcal{P}(p\varepsilon, \mathcal{C}_1 \cup \mathcal{C}_2)$ is equal either to 1 or 0, depending on whether $p\varepsilon$ belongs to \mathcal{C}_2 or not, resp. Now let $n \geq 1$, and let β denote the sequence $X_2 \cdots X_n$. The set $Run(pX_1\beta, \mathcal{C}_1 \cup \mathcal{C}_2)$ is equal to

$$\bigsqcup_{w \in FPath(pX_1\beta, \mathcal{C}_1 \cup \mathcal{C}_2)} Run(w)$$

Let $\mathcal{C}' = \{q\alpha\beta \mid q \in Q, \alpha \in \Gamma^+\}$. We have that

$$\begin{aligned} FPath(pX_1\beta, \mathcal{C}_1 \cup \mathcal{C}_2) &= FPath(pX_1\beta, \mathcal{C}_1 \cap \mathcal{C}' \cup \mathcal{C}_2 \cap \mathcal{C}') \uplus \\ &\bigsqcup_{q \in Q} FPath(pX_1\beta, (\mathcal{C}_1 \setminus \mathcal{C}_2) \cap \mathcal{C}' \cup \{q\beta\}) \odot FPath(q\beta, \mathcal{C}_1 \cup \mathcal{C}_2) \end{aligned}$$

Now observe that for every *simple* set $\mathcal{C} \subseteq \mathcal{C}(\Delta)$ we have that $\mathcal{C} \cap \mathcal{C}' = \mathcal{C}^\bullet\beta$. Hence, the above equation can be rewritten as follows:

$$\begin{aligned} FPath(pX_1\beta, \mathcal{C}_1 \cup \mathcal{C}_2) &= FPath(pX_1\beta, \mathcal{C}_1^\bullet\beta \cup \mathcal{C}_2^\bullet\beta) \uplus \\ &\bigsqcup_{q \in Q} FPath(pX_1\beta, (\mathcal{C}_1 \setminus \mathcal{C}_2)^\bullet\beta \cup \{q\beta\}) \odot FPath(q\beta, \mathcal{C}_1 \cup \mathcal{C}_2) \end{aligned}$$

Using Lemma 3.5 and Lemma 5.6, we obtain that

$$\begin{aligned} \mathcal{P}(Run(pX_1\beta, \mathcal{C}_1 \cup \mathcal{C}_2)) &= \mathcal{P}(Run(pX_1, \mathcal{C}_1 \cup \mathcal{C}_2^\bullet)) + \\ &\sum_{q \in Q} \mathcal{P}(Run(pX_1\beta, (\mathcal{C}_1 \setminus \mathcal{C}_2) \cup \{q\beta\})) \cdot \mathcal{P}(Run(q\beta, \mathcal{C}_1 \cup \mathcal{C}_2)) \end{aligned}$$

This can be also written as

$$\mathcal{P}(Run(pX_1\beta, \mathcal{C}_1 \cup \mathcal{C}_2)) = [pX_1\bullet] + \sum_{q \in Q} [pX_1q] \cdot \mathcal{P}(Run(q\beta, \mathcal{C}_1 \cup \mathcal{C}_2))$$

Now it suffices to apply induction hypothesis to $\mathcal{P}(Run(q\beta, \mathcal{C}_1 \cup \mathcal{C}_2))$ and restructure the resulting expression. \square

Now we show that the probabilities $[pXq]$, $[pX\bullet]$ form the least solution of an effectively constructible system of quadratic equations. This can be seen as a generalization of a similar result for finite-state systems [HJ94, CY95]. 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 3.8).

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

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

- 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 $\mathcal{F} : [0, 1]^{|\mathcal{V}|} \rightarrow [0, 1]^{|\mathcal{V}|}$ 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 3.7. *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 $\langle pX\bullet \rangle_\mu = [pX\bullet]$.*

Proof. Since \mathcal{F} is monotonic and continuous, it has the least fixed point $\mu = \bigvee_{k=0}^{\infty} \mathcal{F}^k(\vec{0})$, where $\vec{0}$ is the tuple of zeros. One can readily check that the tuple π of all $[pXq]$ and $[pX\bullet]$ probabilities forms a solution of the above system; this is done just by partitioning the associated sets of runs into appropriate disjoint subsets similarly as in the proof of Lemma 3.6. Hence, $\mu \leq \pi$. To prove that also $\pi \leq \mu$, we approximate the $[pXq]$ and $[pX\bullet]$ probabilities in the following way: For each $k \in \mathbb{N}_0$ we define

$$\begin{aligned} \bullet [pXq]^k &= \sum_{w \in FPath^k(pX, C_1 \setminus C_2 \cup \{q\epsilon\})} \mathcal{P}(Run(w)) \\ \bullet [pX\bullet]^k &= \sum_{w \in FPath^k(pX, C_1 \cup C_2^*)} \mathcal{P}(Run(w)) \end{aligned}$$

Let π^k be the tuple of all $[pXq]^k$ and $[pX\bullet]^k$ probabilities. Clearly $\pi = \lim_{k \rightarrow \infty} \pi^k$. By induction on k we prove that $\pi^k \leq \mu$ for each $k \in \mathbb{N}_0$, hence also $\pi \leq \mu$ as needed.

The base case ($k = 0$) follows immediately. We show that if $[pXq]^k \leq \langle pXq \rangle_\mu$ and $[pX\bullet]^k \leq \langle pX\bullet \rangle_\mu$, then also $[pXq]^{k+1} \leq \langle pXq \rangle_\mu$ and $[pX\bullet]^{k+1} \leq \langle pX\bullet \rangle_\mu$. If $pX \notin G_1 \setminus G_2$, then $[pXq]^{k+1} = \langle pXq \rangle_\mu = 0$. Otherwise, by applying the

definitions we obtain

$$\begin{aligned}
[pXq]^{k+1} &= \sum_{pX \xrightarrow{x} rYZ} x \cdot \sum_{w \in FPath^k(rYZ, \mathcal{C}_1 \setminus \mathcal{C}_2 \mathcal{U} \{q\varepsilon\})} \mathcal{P}(\text{Run}(w)) \\
&+ \sum_{pX \xrightarrow{x} rY} x \cdot \sum_{w \in FPath^k(rY, \mathcal{C}_1 \setminus \mathcal{C}_2 \mathcal{U} \{q\varepsilon\})} \mathcal{P}(\text{Run}(w)) \\
&+ \sum_{pX \xrightarrow{x} q\varepsilon} x
\end{aligned}$$

and

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

Since

$$\sum_{w \in FPath^k(rY, \mathcal{C}_1 \setminus \mathcal{C}_2 \mathcal{U} \{q\varepsilon\})} \mathcal{P}(\text{Run}(w)) = [rYq]^k,$$

we have

$$\sum_{w \in FPath^k(rY, \mathcal{C}_1 \setminus \mathcal{C}_2 \mathcal{U} \{q\varepsilon\})} \mathcal{P}(\text{Run}(w)) \leq \langle rYq \rangle_\mu$$

by induction hypothesis. Further,

$$\sum_{pX \xrightarrow{x} rYZ} x \cdot \sum_{w \in FPath^k(rYZ, \mathcal{C}_1 \setminus \mathcal{C}_2 \mathcal{U} \{q\varepsilon\})} \mathcal{P}(\text{Run}(w))$$

is surely bounded by

$$\sum_{pX \xrightarrow{x} rYZ} x \cdot \sum_{t \in Q} [rYt]^k \cdot [tZq]^k,$$

which is bounded by

$$\sum_{pX \xrightarrow{x} rYZ} x \cdot \sum_{t \in Q} \langle rYt \rangle_\mu \cdot \langle tZq \rangle_\mu$$

by induction hypothesis. To sum up, we have that $[pXq]^{k+1} \leq \langle pXq \rangle_\mu$. The inequality $[pX\bullet]^{k+1} \leq \langle pX\bullet \rangle_\mu$ is proved similarly. \square

Example 3.8. 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 $\langle pX p \rangle$, 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 3.6 we further obtain that, e.g., $\mathcal{P}(IIZ, \mathcal{C}_1 \cup \mathcal{C}_2) = \min\{1, (1-x)^2/x^2\}$.

In Example 3.8, 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 3.9. Let $\text{Const} = \mathbb{Q} \cup \{[pXq], [pX\bullet] \mid p, q \in Q \text{ and } X \in \Gamma\}$, 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 3.7, $E_1 \sim E_2$ is effectively expressible as a closed formula of $(\mathbb{R}, +, *, \leq)$. Hence, the theorem follows from the decidability of first-order arithmetic of reals [Tar51].

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:

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

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. \square

An immediate consequence of Theorem 3.9 is the following:

Input: $pX \in \mathcal{C}(\Delta)$, $0 < \lambda < 1$

Output: $\mathcal{P}^\ell, \mathcal{P}^u$

```

1:  $\mathcal{P}^\ell := 0; \mathcal{P}^u := 1;$ 
2: for  $i = 1$  to  $\lceil -\log_2 \lambda \rceil$ 
3:   if  $[pX\bullet] + \sum_{q \in \mathcal{C}_2} [pXq] \geq (\mathcal{P}^u - \mathcal{P}^\ell)/2$ 
4:     then  $\mathcal{P}^\ell := (\mathcal{P}^u + \mathcal{P}^\ell)/2$ 
5:     else  $\mathcal{P}^u := (\mathcal{P}^u + \mathcal{P}^\ell)/2$ 
6:   fi

```

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

Theorem 3.10. *Let $p\alpha \in \mathcal{C}(\Delta)$, $\varrho \in [0, 1]$, $\sim \in \{\leq, <, \geq, >\}$ and $0 < \lambda < 1$. It is decidable whether $\mathcal{P}(p\alpha, \mathcal{C}_1 \cup \mathcal{C}_2) \sim \varrho$. Moreover, there effectively exist rational numbers $\mathcal{P}^\ell, \mathcal{P}^u$ such that $\mathcal{P}^\ell \leq \mathcal{P}(p\alpha, \mathcal{C}_1 \cup \mathcal{C}_2) \leq \mathcal{P}^u$ 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 $\mathcal{P}(pX, \mathcal{C}_1 \cup \mathcal{C}_2) \sim \varrho$ iff $[pX\bullet] + \sum_{q \in \mathcal{C}_2} [pXq] \sim \varrho$ by Lemma 3.6. Hence, we can apply Theorem 3.9. The numbers $\mathcal{P}^\ell, \mathcal{P}^u$ are computable, e.g., by the algorithm of Fig. 2. \square

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. \square

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

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

Using Lemma 3.6, we can easily check that $\bigcup_{i=0}^{\infty} S_i = \{p\alpha \in \mathcal{C}(\Delta) \mid \mathcal{P}(p\alpha, \mathcal{C}_1 \cup \mathcal{C}_2) = 1\}$. 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 3.9. It is straightforward to check that $L(\mathcal{M}_p) = \{\alpha \in \Gamma^* \mid p\alpha \in \bigcup_{i=0}^{\infty} S_i\}$. \square

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

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

The fact $\bigcup_{i=0}^{\infty} S_i = \{p\alpha \in \mathcal{C}(\Delta) \mid \mathcal{P}(p\alpha, \mathcal{C}_1 \cup \mathcal{C}_2) = 0\}$ follows immediately from Lemma 3.6. 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. \square

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 $\varphi \equiv \text{tt}$ 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). \square

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 $\mathcal{T} = (S, \rightarrow, Prob)$ 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 $\llbracket \varphi \rrbracket_{\lambda}^{\nu}$, in the same way as $\llbracket \varphi \rrbracket^{\nu}$. The only exception is $\varphi_1 \cup \sim^{\ell} \varphi_2$ where

- if $\sim \in \{<, \leq\}$, then $\llbracket \varphi_1 \mathcal{U}^{\sim \varrho} \varphi_2 \rrbracket_\lambda^\nu = \{s \in S \mid \mathcal{P}(\text{Run}(s, \llbracket \varphi_1 \rrbracket_\lambda^\nu \mathcal{U} \llbracket \varphi_2 \rrbracket_\lambda^\nu)) \sim \varrho + \lambda\}$
- if $\sim \in \{>, \geq\}$, then $\llbracket \varphi_1 \mathcal{U}^{\sim \varrho} \varphi_2 \rrbracket_\lambda^\nu = \{s \in S \mid \mathcal{P}(\text{Run}(s, \llbracket \varphi_1 \rrbracket_\lambda^\nu \mathcal{U} \llbracket \varphi_2 \rrbracket_\lambda^\nu)) \sim \varrho - \lambda\}$

Note that for every negation-free formula φ we have that $\llbracket \varphi \rrbracket^\nu \subseteq \llbracket \varphi \rrbracket_\lambda^\nu$. 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 $s \in \llbracket \varphi \rrbracket^\nu$, then the answer is YES;
- if the answer is YES, then $s \in \llbracket \varphi \rrbracket_\lambda^\nu$.

For the rest of this section, let us fix a pBPA $\Delta = (\Gamma, \delta, \text{Prob})$. 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.)

Proof. We describe just the construction of \mathcal{A}^\geq (the Δ -automaton \mathcal{A}^\leq is constructed similarly). Let $S = \{X \in \Gamma \mid [X, \varepsilon] \neq 1\}$. For every $\alpha \in \Gamma^*$, let $\alpha(S)$ be the string obtained by deleting all symbols of $\Gamma \setminus S$ from α . For each $\beta \in S^*$ we define the set $Cl(\beta) = \{\alpha \in \Gamma^* \mid \alpha(S) = \beta\}$. It follows directly from Lemma 3.6 that for all $\beta \in S^*$ and $\alpha \in Cl(\beta)$ we have that $\mathcal{P}(\beta, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2) = \mathcal{P}(\alpha, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2)$. Further, for all $n \in \mathbb{N}_0$ and $\beta \in \bigcup_{i=0}^n S^n$ we define the set

$$\text{Gen}_n(\beta) = \begin{cases} Cl(\beta) & \text{if } \alpha \in S^i \wedge i < n \\ \{\alpha\alpha' \mid \alpha \in Cl(\beta), \alpha' \in \Gamma^*\} & \text{if } \alpha \in S^n \end{cases}$$

Input: pBPA Δ , $0 < \lambda < 1$
Output: $n, \kappa, \nu, [X, \bullet]^\ell, [X, \varepsilon]^\ell, [X, \bullet]^u, [X, \varepsilon]^u$

- 1: $S := \{X \in \Gamma \mid [X, \varepsilon] \neq 1\}$;
- 2: $\nu := 1; n := \infty$;
- 3: **for each** $X \in S$ **do**
- 4: $[X, \varepsilon]^\ell := 0; [X, \bullet]^\ell := 0; [X, \varepsilon]^u := 1; [X, \bullet]^u := 1$;
- 5: **done**
- 6: **repeat**
- 7: **for each** $X \in \Gamma$ **do**
- 8: $\text{avg}^\varepsilon := ([X, \varepsilon]^u - [X, \varepsilon]^\ell)/2$;
- 9: $\text{avg}^\bullet := ([X, \bullet]^u - [X, \bullet]^\ell)/2$;
- 10: **if** $[X, \varepsilon] \geq \text{avg}^\varepsilon$ **then** $[X, \varepsilon]^\ell := \text{avg}^\varepsilon$;
- 11: **else** $[X, \varepsilon]^u := \text{avg}^\varepsilon$;
- 12: **if** $[X, \bullet] \geq \text{avg}^\bullet$ **then** $[X, \bullet]^\ell := \text{avg}^\bullet$;
- 13: **else** $[X, \bullet]^u := \text{avg}^\bullet$;
- 14: **done**
- 15: $\nu := \nu/2$;
- 16: $\kappa := \max\{[X, \varepsilon]^u \mid X \in S\}$;
- 17: **if** $\kappa < 1$ **then** $n := \lceil (\log(\lambda/3)/\log \kappa) \rceil$
- 18: **until** $\kappa < 1$ **and** $n(\nu + \nu(n+1)(1+\nu)^n) \leq \lambda/3$

Figure 3: A part of the algorithm for pBPA

We prove that for every $0 < \lambda < 1$ there effectively exist $n \in \mathbb{N}_0$ and $\mathcal{G} \subseteq \bigcup_{i=0}^n S^i$ such that for every $\alpha \in \Gamma^*$ we have that

- if $\mathcal{P}(\alpha, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2) \geq \varrho$, then $\alpha \in \bigcup_{\beta \in \mathcal{G}} \text{Gen}_n(\beta)$;
- if $\alpha \in \bigcup_{\beta \in \mathcal{G}} \text{Gen}_n(\beta)$, then $\mathcal{P}(\alpha, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2) \geq \varrho - \lambda$.

This suffices for our purposes, because the set $\bigcup_{\beta \in \mathcal{G}} \text{Gen}_n(\beta)$ is clearly recognizable by an effectively constructible Δ -automaton \mathcal{A}^\geq .

The crucial part of the algorithm for computing the set \mathcal{G} is shown in Fig. 3. The algorithm starts by computing the set S (note that S is effectively computable due to Theorem 3.9). For each $X \in S$, there are four rational variables $[X, \varepsilon]^\ell$, $[X, \varepsilon]^u$, $[X, \bullet]^\ell$, and $[X, \bullet]^u$ whose values are lower and upper approximations of the probabilities $[X, \varepsilon]$ and $[X, \bullet]$, resp. These variables are initialized in lines 3–5 and successively refined in lines 7–14. Note that the conditions of the **if** statements in lines 10 and 12 are effective due to Theorem 3.9. The current “precision”, i.e., the difference between the upper and the lower approximation is stored in the rational variable ν . The subtle point is the termination condition. First, one necessary condition for termination is that $\kappa = \max\{[X, \varepsilon]^u \mid X \in S\}$ becomes less than one. This must happen eventually, because $[X, \varepsilon] < 1$ for every $X \in S$. An important observation is that κ can only *decrease* by performing the assignment in line 16. This means that $n = \lceil (\log(\lambda/3)/\log \kappa) \rceil$ also only *decreases* (since both λ and κ are less than 1, we have $\log(\lambda/3)/\log \kappa = |\log(\lambda/3)|/|\log \kappa|$; and if $0 < \kappa' < \kappa < 1$, then $|\log \kappa'| > |\log \kappa|$). Therefore, we eventually find a sufficiently small ν such that $n(\nu + \nu(n+1)(1+\nu)^n) \leq \lambda/3$.

The output of the algorithm of Fig. 3 are the (values of the) variables $n, \nu, \kappa, [X, \varepsilon]^\ell, [X, \varepsilon]^u, [X, \bullet]^\ell$, and $[X, \bullet]^u$ where X ranges over S . For each $\beta \in S^*$, let $\mathcal{P}^\ell(\beta, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2)$ and $\mathcal{P}^u(\beta, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2)$ be the lower and upper approximations of $\mathcal{P}(\beta, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2)$ obtained by using the formula of Lemma 3.6 where $[X, \varepsilon]^\ell, [X, \bullet]^\ell$, and $[X, \varepsilon]^u, [X, \bullet]^u$ are used instead of $[X, \varepsilon], [X, \bullet]$, respectively. The set \mathcal{G} is constructed as follows:

$$\begin{aligned} \mathcal{G} &= \{\beta \in S^i \mid 0 \leq i < n, \mathcal{P}^u(\beta, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2) \geq \varrho\} \\ &\cup \{\beta \in S^n \mid \mathcal{P}^u(\beta, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2) \geq \varrho - \lambda/3\} \end{aligned}$$

To verify that the set \mathcal{G} has the properties mentioned above, we need to formulate two auxiliary observations.

- (a) for all $\beta \in S^m$ and $\alpha \in \Gamma^*$ we have that

$$|\mathcal{P}(\beta, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2) - \mathcal{P}(\beta\alpha, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2)| \leq \lambda/3$$

This follows immediately from the following (in)equalities:

$$\begin{aligned}\mathcal{P}(\beta\alpha, \mathcal{C}_1 \cup \mathcal{C}_2) &= \mathcal{P}(\beta, \mathcal{C}_1 \cup \mathcal{C}_2^\bullet) + \mathcal{P}(\beta, \mathcal{C}_1 \setminus \mathcal{C}_2 \cup \{\varepsilon\}) \cdot \mathcal{P}(\alpha, \mathcal{C}_1 \cup \mathcal{C}_2) \\ \mathcal{P}(\beta, \mathcal{C}_1 \cup \mathcal{C}_2) &\leq \mathcal{P}(\beta, \mathcal{C}_1 \cup \mathcal{C}_2^\bullet) + \mathcal{P}(\beta, \mathcal{C}_1 \setminus \mathcal{C}_2 \cup \{\varepsilon\}) \\ \mathcal{P}(\beta, \mathcal{C}_1 \setminus \mathcal{C}_2 \cup \{\varepsilon\}) &\leq \lambda/3\end{aligned}$$

The first two (in)equalities are obtained just by applying Lemma 3.6. The last one is derived as follows: $\mathcal{P}(\beta, \mathcal{C}_1 \setminus \mathcal{C}_2 \cup \{\varepsilon\})$ is surely bounded by κ^n (by Lemma 3.6 and the definition of κ). Since $n = \lceil (\log(\lambda/3)/\log \kappa) \rceil$, we have $n \cdot \log \kappa \leq \log(\lambda/3)$. Hence, $\log \kappa^n \leq \log(\lambda/3)$, thus $\kappa^n \leq \lambda/3$.

(b) for each $\beta \in \bigcup_{i=0}^n S^i$ we have that

$$\mathcal{P}^u(\beta, \mathcal{C}_1 \cup \mathcal{C}_2) - \mathcal{P}(\beta, \mathcal{C}_1 \cup \mathcal{C}_2) \leq \lambda/3$$

Let $k = \text{length}(\beta)$. A straightforward induction on k reveals that $\mathcal{P}^u(\beta, \mathcal{C}_1 \cup \mathcal{C}_2) \leq (k+1) \cdot (1+\nu)^k$. Now we prove (again by induction on k) that

$$\mathcal{P}^u(\beta, \mathcal{C}_1 \cup \mathcal{C}_2) - \mathcal{P}(\beta, \mathcal{C}_1 \cup \mathcal{C}_2) \leq k(\nu + \nu(k+1)(1+\nu)^k)$$

The base case (when $k = 0$) is immediate, because $\mathcal{P}^u(\varepsilon, \mathcal{C}_1 \cup \mathcal{C}_2) = \mathcal{P}(\varepsilon, \mathcal{C}_1 \cup \mathcal{C}_2)$. Now let $\beta = X\beta'$. By definition, $\mathcal{P}^u(X\beta', \mathcal{C}_1 \cup \mathcal{C}_2) - \mathcal{P}(X\beta', \mathcal{C}_1 \cup \mathcal{C}_2)$ is equal to

$$[X, \bullet]^u + [X, \varepsilon]^u \cdot \mathcal{P}^u(\beta', \mathcal{C}_1 \cup \mathcal{C}_2) - ([X, \bullet] + [X, \varepsilon] \cdot \mathcal{P}(\beta', \mathcal{C}_1 \cup \mathcal{C}_2)) \quad (1)$$

Since $[X, \bullet]^u \leq [X, \bullet] + \nu$ and $[X, \varepsilon]^u \leq [X, \varepsilon] + \nu$, the expression (1) is bounded by

$$\nu + [X, \varepsilon] \cdot (\mathcal{P}^u(\beta', \mathcal{C}_1 \cup \mathcal{C}_2) - \mathcal{P}(\beta', \mathcal{C}_1 \cup \mathcal{C}_2)) + \nu \cdot \mathcal{P}^u(\beta', \mathcal{C}_1 \cup \mathcal{C}_2) \quad (2)$$

By applying induction hypothesis and the facts that $[X, \varepsilon] \leq 1$ and $\mathcal{P}^u(\beta', \mathcal{C}_1 \cup \mathcal{C}_2) \leq (k+1) \cdot (1+\nu)^k$ (see above), we obtain that the expression (2) is bounded by

$$\nu + k(\nu + \nu(k+1)(1+\nu)^k) + \nu(k+1)(1+\nu)^k$$

which is bounded by $(k+1)(\nu + \nu(k+2)(1+\nu)^{k+1})$ as required. This finishes the inductive step.

Since $n(\nu + \nu(n+1)(1+\nu)^n) \leq \lambda/3$ and $k \leq n$, we have $\mathcal{P}^u(\beta, \mathcal{C}_1 \cup \mathcal{C}_2) - \mathcal{P}(\beta, \mathcal{C}_1 \cup \mathcal{C}_2) \leq k(\nu + \nu(k+1)(1+\nu)^k) \leq \lambda/3$.

Now we are ready to prove that the set \mathcal{G} has the required properties. Let $\alpha \in \Gamma^*$ such that $\mathcal{P}(\alpha, \mathcal{C}_1 \cup \mathcal{C}_2) \geq \varrho$, and let $\beta = \alpha(S)$. There are two possibilities:

- $length(\beta) < n$. Then $\mathcal{P}^u(\beta, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2) \geq \varrho$, hence $\beta \in \mathcal{G}$ and $\alpha \in \bigcup_{\beta \in \mathcal{G}} Gen_n(\beta)$.
- $length(\beta) \geq n$. Let $\beta = \gamma\gamma'$ where $length(\gamma) = n$. Due to the observation (a) above we have that $\mathcal{P}(\gamma, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2) \geq \varrho - \lambda/3$, hence also $\mathcal{P}^u(\gamma, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2) \geq \varrho - \lambda/3$, which means that $\gamma \in \mathcal{G}$ and thus $\alpha \in \bigcup_{\beta \in \mathcal{G}} Gen_n(\beta)$.

Now let $\alpha \in Gen_n(\beta)$ for some $\beta \in \mathcal{G}$. Again, we distinguish two possibilities:

- $length(\beta) < n$. Then $\mathcal{P}^u(\beta, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2) \geq \varrho$, which means that $\mathcal{P}(\beta, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2) \geq \varrho - \lambda/3$ by the observation (b) above. Hence, $\mathcal{P}(\alpha, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2) \geq \varrho - \lambda/3$.
- $length(\beta) = n$. Then $\mathcal{P}^u(\beta, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2) \geq \varrho - \lambda/3$, which means that $\mathcal{P}(\beta, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2) \geq \varrho - 2\lambda/3$ due to the observation (b). Further, for every $\alpha' \in \Gamma$ we have that $\mathcal{P}(\beta\alpha', \mathcal{C}_1 \mathcal{U} \mathcal{C}_2) \geq \varrho - \lambda$ due to the observation (a) above. Hence, $\mathcal{P}(\alpha, \mathcal{C}_1 \mathcal{U} \mathcal{C}_2) \geq \varrho - \lambda$ as required.

The automaton \mathcal{A}^{\leq} is constructed similarly. Here, the set \mathcal{G} is computed using the lower approximations $[X, \bullet]^\ell$ and $[X, \varepsilon]^\ell$. Since this construction is analogous to the one just presented, it is not given explicitly. \square

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). \square

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, $\varrho: B \times \Sigma \rightarrow B$ is a (total) transition function (we write $b \xrightarrow{\alpha} b'$ instead of $\varrho(b, \alpha) = b'$), 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 $b_0b_1 \dots$ of states such that for every $i \geq 0$ there is a $\alpha \in \Sigma$ such that $b_i \xrightarrow{\alpha} b_{i+1}$. A run $b_0b_1 \dots$ 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 [KV98]. Our results can be extended to atomic propositions interpreted over arbitrary regular sets of configurations using the same technique as in [EKS03].

Definition 5.3. *The product of Δ and \mathcal{B} is a probabilistic pushdown automaton $\Delta\mathcal{B} = (Q \times B, \Gamma, \delta', \text{Prob}')$ 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}_{Δ} 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}_{Δ} (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 $b \in \text{Acc}$. A run in $\mathcal{T}_{\Delta\mathcal{B}}$ is accepting if it visits accepting configurations infinitely often. A run in \mathcal{T}_{Δ} 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 $\mathcal{P}(p\alpha, \mathcal{B}) = \mathcal{P}(\{w \in \text{Run}(p\alpha) \mid w \text{ is accepting}\})$.

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 [CY95]). 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 [CY95]. 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 $w = p_0\alpha_0, p_1\alpha_1, \dots$ be an infinite run in \mathcal{T}_Δ . 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.

Now we formulate two auxiliary lemmas which will be used in the proof of Lemma 5.8.

For every finite or infinite path $w = p_1\alpha_1 p_2\alpha_2 \cdots$ in \mathcal{T}_Δ and every $\beta \in \Gamma^*$, the symbol $w^{+\beta}$ denotes the path $p_1\alpha_1\beta p_2\alpha_2\beta \cdots$ obtained from w by concatenating β to the stack content in every configuration. Similarly, if R is a set of paths in \mathcal{T}_Δ and $\beta \in \Gamma^*$, then $[R]^{+\beta}$ denotes the set $\{w^{+\beta} \mid w \in R\}$.

Lemma 5.6. *Let $pX \in Q \times \Gamma$ and $\beta \in \Gamma^*$. Then $\mathcal{P}([IRun(pX)]^{+\beta}) = \mathcal{P}(IRun(pX))$.*

Proof. Let $Dead = Q \times \{\varepsilon\} \cup \{qY\alpha \mid qY \text{ has no transitions in } \delta, \alpha \in \Gamma^*\}$. We have that

$$\begin{aligned} \mathcal{P}([IRun(pX)]^{+\beta}) &= 1 - \mathcal{P}(Run(pX\beta, \mathcal{C}(\Delta)^\bullet \beta \mathcal{U} Dead \beta)) \\ &= 1 - \mathcal{P}(Run(pX, \mathcal{C}(\Delta) \mathcal{U} Dead)) \quad (\text{by Lemma 3.5}) \\ &= \mathcal{P}(IRun(pX)). \end{aligned}$$

□

Lemma 5.7. *Let $s_0 \cdots s_n$ be a path in a probabilistic transition system, and let X be a measurable subset of $Run(s_n)$. Then $\{s_0 \cdots s_n\} \odot X$ is a measurable subset of $Run(s_0)$, and moreover $\mathcal{P}(\{s_0 \cdots s_n\} \odot X) = \prod_{i=1}^n x_i \cdot \mathcal{P}(X)$, where $s_i \xrightarrow{x_{i+1}} s_{i+1}$ for every $0 \leq i < n$. (The ‘ \odot ’ operator has been introduced in Definition 3.3.)*

Proof. Standard. □

Lemma 5.8. *For all $pX \in \mathcal{H}(\Delta)$, $n \in \mathbb{N}$, and v_1, \dots, 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 $\mathcal{P}(V_{pX}^{(1)}=v_1 \wedge \cdots \wedge V_{pX}^{(n)}=v_n) = y$.*

Proof. For the sake of clarity, assume for the moment that the possible values of $V_{pX}^{(i)}$ are \perp or the heads qY of Δ , instead of pairs (qY, f) . At the end of the proof we show how to modify it in order to take into account the boolean flag f .

Let $R \subseteq \text{Run}(pX)$ be a set of runs. We say that $\mathcal{P}(R)$ is *well-definable* if $\mathcal{P}(R)$ is effectively definable from a finite family of probabilities of the form $\mathcal{P}(\text{Run}(qY, \mathcal{C}_1 \cup \mathcal{C}_2))$, where $\mathcal{C}_1, \mathcal{C}_2$ are simple sets, using only summation, multiplication, and rational constants. Note that if $\mathcal{P}(R)$ is well-definable, it can be expressed in $(\mathbb{R}, +, *, \leq)$ using the results of Section 3.

By induction on n we prove that $\mathcal{P}(V_{pX}^{(1)}=v_1 \wedge \cdots \wedge V_{pX}^{(n)}=v_n)$ is well-definable. The base case when $n = 1$ follows immediately, because $\mathcal{P}(V_{pX}^{(1)}=v_1)$ equals either $\mathcal{P}(\text{IRun}(pX))$, $1 - \mathcal{P}(\text{IRun}(pX))$, or 0, depending on whether $v_1 = pX$, $v_1 = \perp$, or $pX \neq v_1 \neq \perp$, respectively. Observe that $\mathcal{P}(\text{IRun}(pX)) = 1 - \mathcal{P}(\text{Run}(pX, \mathcal{C}(\Delta) \cup \text{Dead}))$, where $\text{Dead} = Q \times \{\varepsilon\} \cup \{qY\alpha \mid qY \text{ has no transitions in } \delta, \alpha \in \Gamma^*\}$.

Now let $n \geq 2$. For each $1 \leq i \leq n$, let Sat_i be the set of all runs that satisfy $V_{pX}^{(1)}=v_1 \wedge \cdots \wedge V_{pX}^{(i)}=v_i$. If $\mathcal{P}(\text{Sat}_{n-1}) = 0$, which is decidable by induction hypothesis, then $\mathcal{P}(\text{Sat}_n) = 0$ as well. If $\mathcal{P}(\text{Sat}_{n-1}) \neq 0$ and there is $i \leq n - 1$ such that $v_i = \perp$, then for all $j \leq n - 1$ we have that $v_j = \perp$, and $\mathcal{P}(\text{Sat}_n)$ is equal either to $\mathcal{P}(\text{Sat}_{n-1})$ or 0, depending on whether $v_n = \perp$ or not, respectively. If $\mathcal{P}(\text{Sat}_{n-1}) \neq 0$, $v_i \neq \perp$ for all $i \leq n - 1$, and $v_n = \perp$, then $\mathcal{P}(\text{Sat}_n) = 0$. So, the only interesting case is when $\mathcal{P}(\text{Sat}_{n-1}) \neq 0$ and $v_i \neq \perp$ for all $i \leq n$. Since

$$\mathcal{P}(\text{Sat}_n) = \frac{\mathcal{P}(V_{pX}^{(n)}=v_n \mid \text{Sat}_{n-1})}{\mathcal{P}(\text{Sat}_{n-1})}$$

and $\mathcal{P}(\text{Sat}_{n-1})$ is well-definable by induction hypothesis, it suffices to show that the conditional probability $\mathcal{P}(V_{pX}^{(n)}=v_n \mid \text{Sat}_{n-1})$ is also well-definable. For this we use a general result of basic probability theory saying that if A, B are events and $B = \uplus_{i \in I} B_i$, where I is a finite or countably infinite index set, then

$$\mathcal{P}(A \mid B) = \frac{\sum_{i \in I} \mathcal{P}(A \mid B_i) \cdot \mathcal{P}(B_i)}{\mathcal{P}(B)}$$

An immediate consequence of this equation is that if the probability $\mathcal{P}(A \mid B_i)$ is independent of i , then $\mathcal{P}(A \mid B) = \mathcal{P}(A \mid B_i)$. In our case, A is the event $V_{pX}^{(n)}=v_n$, and B is Sat_{n-1} . Let

$$\text{Chop} = \{w(0) \cdots w(\min_{n-1}(w)) \mid w \in \text{Sat}_{n-1}\}.$$

Observe that if $y \in \text{Chop}$, then the last configuration of y is of the form $p_{n-1}X_{n-1}\alpha$. We denote the α by $\text{Stack}(y)$. For every $y \in \text{Chop}$, let

$$\text{Sat}_{n-1}(y) = \{y\} \odot [\text{IRun}(p_{n-1}X_{n-1})]^{+\text{Stack}(y)} \quad (3)$$

Now we can easily check that

$$Sat_{n-1} = \bigsqcup_{y \in Chop} Sat_{n-1}(y)$$

Hence, *Chop* plays the role of *I*, and $Sat_{n-1}(y)$ plays the role of B_i . We show that $\mathcal{P}(V_{pX}^{(n)}=v_n \mid Sat_{n-1}(y))$ is independent of y , which means that

$$\mathcal{P}(V_{pX}^{(n)}=v_n \mid Sat_{n-1}(y)) = \mathcal{P}(V_{pX}^{(n)}=v_n \mid Sat_{n-1}).$$

By definition of conditional probability,

$$\mathcal{P}(V_{pX}^{(n)}=v_n \mid Sat_{n-1}(y)) = \frac{\mathcal{P}(V_{pX}^{(n)}=v_n \wedge Sat_{n-1}(y))}{\mathcal{P}(Sat_{n-1}(y))} \quad (4)$$

The denominator of the fraction in equation (4) is well-definable, because

$$\mathcal{P}(Sat_{n-1}(y)) = \mathcal{P}(Run(y)) \cdot \mathcal{P}(IRun(p_{n-1}X_{n-1}))$$

Here we used Lemma 5.6, Lemma 5.7, and equation (3). Now we show that $\mathcal{P}(V_{pX}^{(n)}=v_n \wedge Sat_{n-1}(y))$ is also well-definable. Let R be the set of all runs satisfying $V_{pX}^{(n)}=v_n \wedge Sat_{n-1}(y)$, and let $\alpha = Stack(y)$. Obviously, each $w \in R$ starts with y . Now let us consider what transitions can be performed from the final state $p_{n-1}X_{n-1}\alpha$ of y .

- Obviously, transitions which decrease the stack cannot be performed, because $p_{n-1}X_{n-1}\alpha$ would not be a minimum then (i.e., w would not belong to R).
- If a transition of the form $p_{n-1}X_{n-1}\alpha \xrightarrow{x} rZ\alpha$ is performed, then $rZ\alpha$ must be the n -th minimum, because the stack cannot be decreased below Z (otherwise, $p_{n-1}X_{n-1}\alpha$ would not be a minimum). So, if $w \in R$, we must have that $rZ = p_nX_n$.
- If a transition of the form $p_{n-1}X_{n-1}\alpha \xrightarrow{x} rPQ\alpha$ is performed, then the stack cannot be decreased below Q . Now there are two possibilities:
 - If the stack is never decreased below P , then the configuration $rPQ\alpha$ is the n -th minimum. Hence, if $w \in R$, we must have that $rP = p_nX_n$.
 - If the stack is decreased below P , i.e., if a sequence of transitions of the form $rPQ\alpha \rightarrow^* tQ\alpha$, where $rP \rightarrow^* t\varepsilon$, is performed, then $tQ\alpha$ is the n -th minimum. Hence, if $w \in R$, we must have that $tQ = p_nX_n$.

From the above discussion, it follows that R can be partitioned as follows:

$$\begin{aligned}
R = & \biguplus_{p_{n-1}X_{n-1} \xrightarrow{x} p_n X_n} \{y\} \odot \{p_{n-1}X_{n-1}\alpha \ p_n X_n \alpha\} \odot [IRun(p_n X_n)]^{+\alpha} \\
& \biguplus_{\substack{p_{n-1}X_{n-1} \xrightarrow{x} p_n X_n Y \\ Y \in \Gamma}} \{y\} \odot \{p_{n-1}X_{n-1}\alpha \ p_n X_n Y \alpha\} \odot [IRun(p_n X_n)]^{+Y\alpha} \\
& \biguplus_{\substack{p_{n-1}X_{n-1} \xrightarrow{x} qY X_n \\ q \in Q, Y \in \Gamma}} \{y\} \odot [FPath(qY, \mathcal{C}(\Delta) \mathcal{U} \{p_n \varepsilon\})]^{+\alpha} \odot [IRun(p_n X_n)]^{+\alpha}
\end{aligned}$$

Using Lemma 5.6, Lemma 5.7, and the above equation, we obtain that

$$\mathcal{P}(V_{pX}^{(n)} = v_n \wedge Sat_{n-1}(y)) = \mathcal{P}(Run(y)) \cdot \mathcal{P}(IRun(p_n X_n)) \cdot S$$

where

$$\begin{aligned}
S = & \sum_{p_{n-1}X_{n-1} \xrightarrow{x} p_n X_n} x + \sum_{\substack{p_{n-1}X_{n-1} \xrightarrow{x} p_n X_n Y \\ Y \in \Gamma}} x + \\
& \sum_{\substack{p_{n-1}X_{n-1} \xrightarrow{x} qY X_n \\ q \in Q, Y \in \Gamma}} x \cdot \mathcal{P}(Run(qY, \mathcal{C}(\Delta) \mathcal{U} \{p_n \varepsilon\})) \quad (5)
\end{aligned}$$

Equation (4) can now be rewritten to

$$\mathcal{P}(V_{pX}^{(n)} = v_n \mid Sat_{n-1}(y)) = \frac{\mathcal{P}(IRun(p_n X_n))}{\mathcal{P}(IRun(p_{n-1} X_{n-1}))} \cdot S \quad (6)$$

where the meaning of S is given by equation (5). So, $\mathcal{P}(V_{pX}^{(n)} = v_n \mid Sat_{n-1}(y))$ is indeed independent of y , and hence equation (6) also defines the probability $\mathcal{P}(V_{pX}^{(n)} = v_n \mid Sat_{n-1})$.

We still have to consider the case in which the v_i 's are pairs of the form $(p_i X_i, f_i)$. For this we can use the same construction as above with a minor modification in equation (5). Let us first consider the case when $v_n = (p_n X_n, 0)$. If p_n is accepting, then $\mathcal{P}(V_{pX}^{(n)} = (p_n X_n, 0) \mid Sat_{n-1})$ is zero. Otherwise, $\mathcal{P}(V_{pX}^{(n)} = (p_n X_n, 0) \mid Sat_{n-1})$ is given by the right-hand side of equation (6) where the following sum S' is used instead of S :

$$\begin{aligned}
S' = & \sum_{p_{n-1}X_{n-1} \xrightarrow{x} p_n X_n} x + \sum_{\substack{p_{n-1}X_{n-1} \xrightarrow{x} p_n X_n Y \\ Y \in \Gamma}} x + \\
& \sum_{\substack{p_{n-1}X_{n-1} \xrightarrow{x} qY X_n \\ q \in Q, Y \in \Gamma}} x \cdot \mathcal{P}(Run(qY, (Q \setminus Acc_\Delta) \times \Gamma^+ \mathcal{U} \{p_n \varepsilon\}))
\end{aligned}$$

Finally, let us consider the case when $v_n = (p_n X_n, 1)$. If p_n is accepting, then $\mathcal{P}(V_{pX}^{(n)} = (p_n X_n, 1) \mid \text{Sat}_{n-1}) = 1$. Otherwise, we use the right-hand side of equation (6) where S is replaced with S'' given by

$$S'' = \sum_{\substack{p_{n-1} X_{n-1} \xrightarrow{x} qY X_n \\ q \in Q, Y \in \Gamma}} x \cdot D$$

where

$$D = \mathcal{P}(\text{Run}(qY, \mathcal{C}(\Delta) \cup \{p_n \varepsilon\})) - \mathcal{P}(\text{Run}(qY, (Q \setminus \text{Acc}_\Delta) \times \Gamma^+ \cup \{p_n \varepsilon\}))$$

□

The following lemma proves the Markov property. In fact, it follows immediately from equation (6) used in the proof of Lemma 5.8.

Lemma 5.9. *The conditional probability of $V_{pX}^{(n)} = v_n$ on the hypothesis $V_{pX}^{(1)} = v_1 \wedge \dots \wedge V_{pX}^{(n-1)} = v_{n-1}$ 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 $V_{pX}^{(1)} = v_1 \wedge \dots \wedge V_{pX}^{(n-1)} = v_{n-1}$ 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 $q \in \text{Acc}_\Delta$ or not, respectively. Another consequence of Lemma 5.8 is the following:

Lemma 5.10. *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 5.11. *Let M_Δ be a finite-state Markov chain where the set of states is*

$$\begin{aligned} & \{(qY, 0) \mid q \notin \text{Acc}_\Delta, Y \in \Gamma, \mathcal{P}(V_{qY}^{(1)} = (qY, 0)) > 0\} \\ & \cup \{(qY, 1) \mid qY \in \mathcal{H}(\Delta), \mathcal{P}(V_{qY}^{(1)} = (qY, f_q)) > 0\} \\ & \cup \mathcal{H}(\Delta) \cup \{\perp\} \end{aligned}$$

and transition probabilities are defined as follows:

- $\text{Prob}(\perp \rightarrow \perp) = 1$,
- $\text{Prob}(pX \rightarrow (qY, f)) = \mathcal{P}(V_{pX}^{(1)} = (qY, f))$,

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

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 $\sigma(0) \sigma(1) \dots$ of states of M_Δ where $Prob(\sigma(i) \rightarrow \sigma(i+1)) > 0$ for every $i \in \mathbb{N}_0$.

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

- $\sigma_w(0) = pX$
- if w is finite, then for every $i \in \mathbb{N}$ we have $\sigma_w(i) = \perp$;
- if w is infinite, then for every $i \in \mathbb{N}$ we have $\sigma_w(i) = (p_i X_i, f_i)$, where $p_i X_i$ 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 σ_w is a trajectory in M_Δ . Our next lemma reveals that almost all runs are good.

Lemma 5.12. *Let $pX \in \mathcal{H}(\Delta)$, and let *Good* be the subset of all good runs of $Run(pX)$. Then $\mathcal{P}(\text{Good}) = 1$.*

Proof. Let $Bad = Run(pX) \setminus Good$. Let $Fail$ be the set of all finite sequences $v_0 \dots v_{i+1}$ of states of M_Δ such that $i \in \mathbb{N}_0$, $v_0 = pX$, $v_0 \dots v_i$ is a trajectory in M_Δ , and $Prob(v_i \rightarrow v_{i+1}) = 0$, where $Prob$ is the probability assignment of M_Δ . Each $y \in Fail$ determines a set $Bad_y = \{w \in Bad \mid \sigma_w \text{ starts with } y\}$. Obviously, $Bad = \bigsqcup_{y \in Fail} Bad_y$. We prove that $\mathcal{P}(Bad_y) = 0$ for each $y \in Fail$. Let $y = v_0 \dots v_{i+1}$. By applying definitions, we obtain

$$\begin{aligned} \mathcal{P}(Bad_y) &= \mathcal{P}(V_{pX}^{(1)} = v_1 \wedge \dots \wedge V_{pX}^{(i+1)} = v_{i+1}) \\ &= \frac{\mathcal{P}(V_{pX}^{(i+1)} = v_{i+1} \mid V_{pX}^{(i)} = v_i \wedge \dots \wedge V_{pX}^{(1)} = v_1)}{\mathcal{P}(V_{pX}^{(i)} = v_i \wedge \dots \wedge V_{pX}^{(1)} = v_1)} \end{aligned}$$

Since $\mathcal{P}(V_{pX}^{(i)} = v_i \wedge \dots \wedge V_{pX}^{(1)} = v_1) \neq 0$, the last fraction makes sense and it is equal to

$$\frac{Prob(v_i \rightarrow v_{i+1})}{\mathcal{P}(V_{pX}^{(i)} = v_i \wedge \dots \wedge V_{pX}^{(1)} = v_1)}$$

which equals zero. □

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 $(qY, 0)$ and $(qY, 1)$ either belong to the *same* bottom strongly connected component of M_Δ , or do not belong to a bottom strongly connected component of M_Δ .

Definition 5.13. *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 $f \in \{0, 1\}$ (here the state pX of M_Δ is considered as a root of M_Δ).*

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 $w \in \text{Run}(pX)$ hits a head $qY \in \mathcal{H}(\Delta)$ if there is some $i \in \mathbb{N}$ such that the head of $\text{min}_i(w)$ is qY . The next lemma says that an infinite run eventually hits a recurrent head.

Lemma 5.14. *Let us assume that $\mathcal{P}(\text{IRun}(pX)) > 0$. Then the conditional probability that $w \in \text{Run}(pX)$ hits a recurrent head on the hypothesis that w is infinite is equal to one.*

Proof. Let Rec denote the event that a run of $\text{Run}(pX)$ hits a recurrent head. Due to Lemma 5.12, we have that

$$\mathcal{P}(\text{Rec} \mid \text{IRun}(pX)) = \mathcal{P}(\text{Rec} \mid \text{IRun}(pX) \cap \text{Good}) \quad (7)$$

A run belongs to $\text{IRun}(pX) \cap \text{Good}$ iff its footprint is a trajectory in M_Δ that does not hit the state \perp . A run $w \in \text{IRun}(pX) \cap \text{Good}$ satisfies Rec iff its footprint hits a state of the form (qY, f) , where qY is a recurrent head. It follows directly from the definition of M_Δ that the right-hand side of equation (7) is equal to the probability that a trajectory from pX in M_Δ hits a bottom strongly connected component on the hypothesis that the state \perp is not visited. Since $M(\Delta)$ is finite, this happens with probability one by ergodicity. \square

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

Proof. The argument is similar as in the proof of Lemma 5.14. Let C be a bottom strongly connected component of M_Δ . By ergodicity, the conditional probability that an infinite trajectory in M_Δ hits each state of C infinitely often on the hypothesis that the trajectory hits C is equal to one. \square

A simple consequence of Lemma 5.15 is the following:

Lemma 5.16. (cf. Proposition 4.1.5 of [CY95]) 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 5.17. 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 3.9, 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 5.16 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 [HJ94, CY95] 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

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

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 5.8, 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 5.11 into $(\mathbb{R}, +, *, \leq)$, using the formulae provided by Lemma 5.8). Let $\Psi_{\vec{C}}$ by a conjunction of all these formulae. The formula Φ is constructed as follows:

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

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 [CY95], 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 $p_i X_i$ out of a set $\mathcal{H} = \{p_1 X_1, \dots, p_n X_n\}$ and visit some accepting configuration along the way. We can also decide if *some* head $p_j X_j \in \mathcal{H}(\Delta)$ 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 [Gri88] 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 [CY95], 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

- [ABIJ00] P.A. Abdulla, C. Baier, S.P. Iyer, and B. Jonsson. Reasoning about probabilistic channel systems. In *Proceedings of CONCUR 2000*, volume 1877 of *Lecture Notes in Computer Science*, pages 320–330. Springer, 2000.
- [AEY01] R. Alur, K. Etessami, and M. Yannakakis. Analysis of recursive state machines. In *Proceedings of CAV 2001*, volume 2102 of *Lecture Notes in Computer Science*, pages 207–220. Springer, 2001.
- [AMP99] A. Abney, D. McAllester, and F. Pereira. Relating probabilistic grammars and automata. In *Proceedings of ACP'99*, pages 542–549, 1999.
- [AR03] P.A. Abdulla and A. Rabinovich. Verification of probabilistic systems with faulty communication. In *Proceedings of FoSSaCS 2003*, volume 2620 of *Lecture Notes in Computer Science*, pages 39–53. Springer, 2003.
- [ASB⁺95] 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*, volume 939 of *Lecture Notes in Computer Science*, pages 155–165. Springer, 1995.
- [BE99] 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 Proba-*

bilistic Systems (ARTS'99), volume 1601 of *Lecture Notes in Computer Science*, pages 34–52. Springer, 1999.

- [BS97] O. Burkart and B. Steffen. Model checking the full modal mu-calculus for infinite sequential processes. In *Proceedings of ICALP'97*, volume 1256 of *Lecture Notes in Computer Science*, pages 419–429. Springer, 1997.
- [BS03] N. Bertrand and Ph. Schnoebelen. Model checking lossy channel systems is probably decidable. In *Proceedings of FoSSaCS 2003*, volume 2620 of *Lecture Notes in Computer Science*, pages 120–135. Springer, 2003.
- [BW90] J.C.M. Baeten and W.P. Weijland. *Process Algebra*. Number 18 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1990.
- [CSS03] J.M. Couvreur, N. Saheb, and G. Sutre. An optimal automata approach to LTL model checking of probabilistic systems. In *Proceedings of LPAR 2003*, volume 2850 of *Lecture Notes in Computer Science*, pages 361–375. Springer, 2003.
- [CY88] C. Courcoubetis and M. Yannakakis. Verifying temporal properties of finite-state probabilistic programs. In *Proceedings of 29th Annual Symposium on Foundations of Computer Science*, pages 338–345. IEEE Computer Society Press, 1988.
- [CY95] C. Courcoubetis and M. Yannakakis. The complexity of probabilistic verification. *Journal of the Association for Computing Machinery*, 42(4):857–907, 1995.
- [EHRS00] J. Esparza, D. Hansel, P. Rossmanith, and S. Schwoon. Efficient algorithms for model checking pushdown systems. In *Proceedings of CAV 2000*, volume 1855 of *Lecture Notes in Computer Science*, pages 232–247. Springer, 2000.
- [EKM04] J. Esparza, A. Kučera, and R. Mayr. Model checking probabilistic pushdown automata. Technical Report FIMU-RS-2004-03, Faculty of Informatics, Masaryk University, 2004.
- [EKS03] J. Esparza, A. Kučera, and S. Schwoon. Model-checking LTL with regular valuations for pushdown systems. *Information and Computation*, 186(2):355–376, 2003.

- [Gri88] D. Grigoriev. Complexity of deciding Tarski algebra. *Journal of Symbolic Computation*, 5(1–2):65–108, 1988.
- [HJ94] H. Hansson and B. Jonsson. A logic for reasoning about time and reliability. *Formal Aspects of Computing*, 6:512–535, 1994.
- [HK97] M. Huth and M.Z. Kwiatkowska. Quantitative analysis and model checking. In *Proceedings of LICS’97*, pages 111–122. IEEE Computer Society Press, 1997.
- [HS84] S. Hart and M. Sharir. Probabilistic temporal logic for finite and bounded models. In *Proceedings of POPL’84*, pages 1–13. ACM Press, 1984.
- [IN97] S.P. Iyer and M. Narasimha. Probabilistic lossy channel systems. In *Proceedings of TAPSOFT’97*, volume 1214 of *Lecture Notes in Computer Science*, pages 667–681. Springer, 1997.
- [KV98] O. Kupferman and M.Y. Vardi. Freedom, weakness, and determinism: From linear-time to branching-time. In *Proceedings of LICS’98*, pages 81–92. IEEE Computer Society Press, 1998.
- [LS82] D. Lehman and S. Shelah. Reasoning with time and chance. *Information and Control*, 53:165–198, 1982.
- [MO98] I. Macarie and M. Ogihara. Properties of probabilistic pushdown automata. *Theoretical Computer Science*, 207:117–130, 1998.
- [Rab03] A. Rabinovich. Quantitative analysis of probabilistic lossy channel systems. In *Proceedings of ICALP 2003*, volume 2719 of *Lecture Notes in Computer Science*, pages 1008–1021. Springer, 2003.
- [Tar51] A. Tarski. *A Decision Method for Elementary Algebra and Geometry*. Univ. of California Press, Berkeley, 1951.
- [Var85] M. Vardi. Automatic verification of probabilistic concurrent finite-state programs. In *Proceedings of 26th Annual Symposium on Foundations of Computer Science*, pages 327–338. IEEE Computer Society Press, 1985.
- [Wal01] I. Walukiewicz. Pushdown processes: Games and model-checking. *Information and Computation*, 164(2):234–263, 2001.

**Copyright © 2004, Faculty of Informatics, Masaryk University.
All rights reserved.**

**Reproduction of all or part of this work
is permitted for educational or research use
on condition that this copyright notice is
included in any copy.**

**Publications in the FI MU Report Series are in general accessible
via WWW and anonymous FTP:**

`http://www.fi.muni.cz/informatics/reports/
ftp ftp.fi.muni.cz (cd pub/reports)`

Copies may be also obtained by contacting:

**Faculty of Informatics
Masaryk University
Botanická 68a
602 00 Brno
Czech Republic**