Probabilistic Büchi Automata with non-extremal acceptance thresholds

Rohit Chadha[1], A. Prasad Sistla[2], and Mahesh Viswanathan[3]

[1] LSV, ENS Cachan & CNRS & INRIA Saclay, France
[2] Univ. of Illinois, Chicago, USA
[3] Univ. of Illinois, Urbana-Champaign, USA

Abstract. This paper investigates the power of Probabilistic Büchi Automata (PBA) when the threshold probability of acceptance is non-extremal, i.e., is a value strictly between 0 and 1. Many practical randomized algorithms are designed to work under non-extremal threshold probabilities and thus it is important to study power of PBAs for such cases.
The paper presents a number of surprising expressiveness and decidability results for PBAs when the threshold probability is non-extremal. Some of these results sharply contrast with the results for extremal threshold probabilities. The paper also presents results for Hierarchical PBAs and for an interesting subclass of them called simple PBAs.

1 Introduction

Probabilistic Büchi Automata (PBA), introduced in [2] to model *open*, *reactive* probabilistic systems, are finite state machines that process input strings of infinite length like Büchi automata. However, unlike Büchi automata, they have probabilistic transitions. The semantics of such machines is defined as follows. A run on an input word is considered to be *accepting* if it satisfies the Büchi acceptance condition. The collection of all accepting runs on any input is known to be measurable [14, 2]. For any given acceptance threshold x, the language $\mathcal{L}_{>x}(\mathcal{B})$ ($\mathcal{L}_{\geq x}(\mathcal{B})$) of a PBA $\mathcal{B}$ is defined to be the set of all inputs for which the above measure is $> x$ ($\geq x$).

In a series of papers [2, 1, 9, 4], researchers have studied the behavior of PBAs when the acceptance threshold x is either 0 or 1, delineating the expressive power of such machines and establishing the precise complexity of various decision problems. While extremal thresholds (of 0 and 1) are important for studying randomized algorithms and protocols, in many practical scenarios only algorithms with non-extremal thresholds can solve the problem — consensus in synchronized distributed systems [15], and semantic security [8], being a couple of examples. Thus, studying PBAs under non-extremal thresholds, which is the focus of this paper, is important.

We begin by observing that for non-extremal thresholds $x \in (0, 1)$, the actual value of x itself is not important: for every PBA $\mathcal{B}$, one can efficiently construct another PBA $\mathcal{B}'$ such that $\mathcal{L}_{>x}(\mathcal{B}) = \mathcal{L}_{>\frac{1}{2}}(\mathcal{B}')$ (or $\mathcal{L}_{\geq x}(\mathcal{B}) = \mathcal{L}_{\geq\frac{1}{2}}(\mathcal{B}')$). Thus, we consider the acceptance threshold to be always $\frac{1}{2}$. Our results on the decidability of the emptiness and universality decision problems are summarized in Figure 1.

A few salient points about our results on decision problems are as follows. Typically, solving decision problems for automata with non-extremal thresholds is harder than for those with extremal thresholds, as is borne out by similar results for probabilistic finite automata [11, 6] and for finite state probabilistic monitors [3]. Interestingly, this observation does not hold for checking emptiness of $\mathcal{L}_{>0}(\mathcal{B})$ for a given PBA $\mathcal{B}$, but holds for other problems. More specifically, for a given PBA $\mathcal{B}$, the problems of checking emptiness of $\mathcal{L}_{>\frac{1}{2}}(\mathcal{B})$ and emptiness of $\mathcal{L}_{>0}(\mathcal{B})$ have the same level of undecidability; both of them are $\mathbf{\Sigma}^0_2$-complete. On the other hand, the problems of checking emptiness and universality of $\mathcal{L}_{\geq\frac{1}{2}}(\mathcal{B})$ are $\mathbf{\Pi}^1_1$-complete and **co-R.E.**-complete, respectively, as opposed to both being **PSPACE**-complete for $\mathcal{L}_{=1}(\mathcal{B})$. The universality problem for $\mathcal{L}_{>\frac{1}{2}}(\mathcal{B})$ is $\mathbf{\Pi}^1_1$-complete as opposed to being $\mathbf{\Sigma}^0_2$-complete for $\mathcal{L}_{>0}(\mathcal{B})$.

Previously, we [4] had introduced a syntactic subclass of PBAs called *hierarchical* PBA*s* (HPBA) as an expressively less powerful, but computationally more tractable fragment of PBAs. With extremal thresholds, the emptiness and universality problems are efficiently decidable — emptiness and universality of $\mathcal{L}_{>0}(\mathcal{B})$ are **NL**-complete and **PSPACE**-complete, respectively, while for $\mathcal{L}_{=1}(\mathcal{B})$ they are **PSPACE**-complete and **NL**-complete, when $\mathcal{B}$ is an HPBA. Considering non-extremal acceptance thresholds, these decision problems not only become undecidable, but are *as difficult as* in the case general PBAs. The only exception to this is the case of checking emptiness of $\mathcal{L}_{>\frac{1}{2}}(\mathcal{B})$ which is **co-R.E.**-complete when $\mathcal{B}$ is an HPBA and is $\mathbf{\Sigma}^0_2$-complete for general PBAs. This upper bound of **co-R.E.** in this case is established by observing that for an HPBA $\mathcal{B}$, $\mathcal{L}_{>\frac{1}{2}}(\mathcal{B})$ is non-empty if and only if there is an ultimately periodic word in $\mathcal{L}_{>\frac{1}{2}}(\mathcal{B})$; this observation may be of independent interest.

Next, our undecidability proofs for these various decision problems rely on Condon and Lipton's [6] ideas, used to show the undecidability of the emptiness problem of probabilistic finite automata. However, in order to obtain lower bounds for HPBAs and obtain "hierarchical" machines, we modify the original reduction by Condon and Lipton, and we believe our modification yields a conceptually simpler proof of the undecidability of the emptiness problem for probabilistic finite automata. In order, to prove the undecidability result, Condon and Lipton do the following. Given a 2-counter machine M, they construct a probabilistic finite automata $\mathcal{A}_M$ whose inputs are computations of M, such that a correct halting computation of M, repeated sufficiently many times, is accepted by $\mathcal{A}_M$ with high probability ($> \frac{1}{2}$) and all other inputs are rejected with high probability. Thus, $\mathcal{L}_{>\frac{1}{2}}(\mathcal{A}_M)$ is non-empty iff M has a halting computation. Now, in order to carry out this reduction, the automaton $\mathcal{A}_M$ "checks" every pair of successive configurations in the input for correctness, and maintains a variety of bounded counters to ensure that the asymptotic probability of acceptance has the desired properties. We observe that if the automaton only "checks" one pair of successive configurations (where the pair to be checked is chosen randomly) the reduction still works, yielding a "simpler" automaton construction and a simpler analysis of the assymptotics. However, one casualty of our simpler proof is the following — while we can show that the emptiness problem of probabilistic finite automata is undecidable, the Condon-Lipton proof establishes a stronger fact, namely, that the problem remains undecidable even under the promise that the acceptance probability of every input is bounded away from $\frac{1}{2}$.

Our next set of results pertain to the expressiveness of PBAs and HPBAs with non-extremal acceptance thresholds. Let $\mathbb{L}(\mathrm{PBA}^{>0})$ be the collection of all languages recognized by PBAs with threshold 0, $\mathbb{L}(\mathrm{PBA}^{=1})$ be those recognized with threshold 1, $\mathbb{L}(\mathrm{PBA}^{>\frac{1}{2}})$ be those recognized with a strict threshold of $\frac{1}{2}$, and $\mathbb{L}(\mathrm{PBA}^{\geq\frac{1}{2}})$ be those recognized with a non-strict threshold of $\frac{1}{2}$. Results in [1,4] establish that $\mathbb{L}(\mathrm{PBA}^{>0})$ is closed under complementation, $\mathbb{L}(\mathrm{PBA}^{=1})$ is not closed under complementation, and $\mathbb{L}(\mathrm{PBA}^{>0})$ is the Boolean closure of $\mathbb{L}(\mathrm{PBA}^{=1})$. Observations in [4] already imply that $\mathbb{L}(\mathrm{PBA}^{\geq\frac{1}{2}})$ is not closed under complementation. Moreover, the complexity results of the decision problems in Theorem 1 imply that if $\mathbb{L}(\mathrm{PBA}^{>\frac{1}{2}})$ were complementable, the procedure would not be recursive. We establish that, in fact, $\mathbb{L}(\mathrm{PBA}^{>\frac{1}{2}})$ is not closed under complementation, and therefore cannot be the Boolean closure of $\mathbb{L}(\mathrm{PBA}^{\geq\frac{1}{2}})$. We also show that even though $\mathbb{L}(\mathrm{PBA}^{\geq\frac{1}{2}})$ is a topologically simpler class of languages than $\mathbb{L}(\mathrm{PBA}^{>\frac{1}{2}})$, it is not contained in $\mathbb{L}(\mathrm{PBA}^{>\frac{1}{2}})$; in fact, the two sets $\mathbb{L}(\mathrm{PBA}^{>\frac{1}{2}})$ and $\mathbb{L}(\mathrm{PBA}^{\geq\frac{1}{2}})$ are incomparable. The classes $\mathbb{L}(\mathrm{HPBA}^{>0})$, $\mathbb{L}(\mathrm{HPBA}^{=1})$, $\mathbb{L}(\mathrm{HPBA}^{>\frac{1}{2}})$, and $\mathbb{L}(\mathrm{HPBA}^{\geq\frac{1}{2}})$ can be analogously defined for HPBAs. It was shown in [4] that HPBAs with extremal thresholds correspond to regular languages — $\mathbb{L}(\mathrm{HPBA}^{=1})$ is exactly the set of deterministic ω-regular languages, while $\mathbb{L}(\mathrm{HPBA}^{>0})$ is exactly the set of ω-regular languages. With non-extremal thresholds, HPBAs can recognize non-regular languages. In addition, the observations about PBA expressiveness extend to HPBAs: $\mathbb{L}(\mathrm{HPBA}^{>\frac{1}{2}})$ and $\mathbb{L}(\mathrm{HPBA}^{\geq\frac{1}{2}})$ are not closed under complementation and they are incomparable.

Our motivation in considering HPBAs in [4] was that with extremal thresholds, they were a "regular", tractable subclass of PBAs. However, as observed in the preceding paragraphs, many of these nice properties of HPBAs are lost when considering non-extremal thresholds. Therefore we consider a syntactic subclass of HPBAs that we call *simple* PBA*s* (SPBA). In simple PBAs, the states are partitioned into two sets. The initial and final states belong to the first partition, and the transitions out of states in the first partition are such that at most one successor belongs to the first partition. Transitions from states in the second partition all remain within the second partition. We show that emptiness and universality problems for such machines is tractable, and that the collection of languages recognized by simple PBAs with strict and non-strict thresholds is exactly the class of deterministic ω-regular languages.

The rest of the paper is organized as follows. Section 2 contains some preliminaries. Section 3 contains some examples motivating HPBAs. Section 4 contains the undecidability results for emptiness and universality of of PBAs and HPBAs. Section 5 contains our expressiveness results. Section 6 contains our results on simple PBAs and we conclude in Section 7. The missing proofs can be found in [5].

2 Preliminaries

We assume that the reader is familiar with arithmetical and analytical hierarchies. We also assume that the reader is familiar with Büchi automata and ω-regular languages.

The set of natural numbers will be denoted by $\mathbb{N}$, the closed unit interval by $[0,1]$ and the open unit interval by $(0,1)$. The power-set of a set X will be denoted by 2^X.

Sequences. Given a finite set S, $|S|$ denotes the cardinality of S. Given a sequence (finite or infinite) $\kappa = s_0 s_1 \dots$ over S, $|\kappa|$ will denote the length of the sequence (for infinite sequence $|\kappa|$ will be ω), and $\kappa[i]$ will denote the ith element s_i of the sequence. As usual S^* will denote the set of all finite sequences/strings/words over S, S^+ will denote the set of all finite non-empty sequences/strings/words over S and S^ω will denote the set of all infinite sequences/strings/words over S. Given $\eta \in S^*$ and $\kappa \in S^* \cup S^\omega$, $\eta\kappa$ is the sequence obtained by concatenating the two sequences in order. Given $\mathsf{L}_1 \subseteq \Sigma^*$ and $\mathsf{L}_2 \subseteq \Sigma^\omega$, the set $\mathsf{L}_1\mathsf{L}_2$ is defined to be $\{\eta\kappa \mid \eta \in \mathsf{L}_1 \text{ and } \kappa \in \mathsf{L}_2\}$. Given natural numbers $i, j \leq |\kappa|$, $\kappa[i : j]$ is the finite sequence $s_i, \dots s_j$ and $\kappa[i : \infty]$ is the infinite sequence $s_i, s_{i+1} \dots$, where $s_k = \kappa[k]$. The set of *finite prefixes* of κ is the set $\mathit{Pref}(\kappa) = \{\kappa[0, j] \mid j \in \mathbb{N}, j \leq |\kappa|\}$.

Languages of infinite words. A language L of infinite words over a finite alphabet Σ is a subset of Σ^ω. (Please note we restrict only to finite alphabets.) A language L is said to be a *safety language* if L is *prefix-closed*, *i.e.*, if for every infinite string α, if every prefix of α is a prefix of some string in L, then α itself is in L.

Probabilistic Büchi Automaton (PBA**).** We recall the definition of PBA given in [2]. Informally, a PBA is like a finite-state deterministic Büchi automaton except that the transition function from a state on a given input is described as a probability distribution which determines the probability of the next state. PBAs generalize the probabilistic finite automata (PFAs) [12, 13, 11] on finite input strings to infinite input strings.

Definition 1. A *finite state probabilistic Büchi automata (*PBA*)* over a finite alphabet Σ is a tuple $\mathcal{B} = (Q, q_s, Q_f, \delta)$ where Q is a finite set of *states*, $q_s \in Q$ is the *initial state*, $Q_f \subseteq Q$ is the set of *accepting/final states*, and $\delta : Q \times \Sigma \times Q \to [0, 1]$ is the *transition relation* such that for all $q \in Q$ and $a \in \Sigma$, $\delta(q, a, q')$ is a rational number and $\sum_{q' \in Q} \delta(q, a, q') = 1$.

Notation: The transition function δ of PBA $\mathcal{B}$ on input a can be seen as a square matrix δ_a of order $|Q|$ with the rows labeled by "current" state, columns labeled by "next state" and the entry $\delta_a(q, q')$ equal to $\delta(q, a, q')$. Given a word $u = a_0 a_1 \dots a_n \in \Sigma^+$, δ_u is the matrix product $\delta_{a_0}\delta_{a_1}\dots\delta_{a_n}$. For an empty word $\epsilon \in \Sigma^*$ we take δ_ϵ to be the identity matrix. Finally for any $Q_0 \subseteq Q$, we say that $\delta_u(q, Q_0) = \sum_{q' \in Q_0} \delta_u(q, q')$. Given a state $q \in Q$ and a word $u \in \Sigma^+$, $\mathsf{post}(q, u) = \{q' \mid \delta_u(q, q') > 0\}$.

Intuitively, the PBA starts in the initial state q_s and if after reading $a_0, a_1 \dots, a_i$ results in state q, then it moves to state q' with probability $\delta_{a_{i+1}}(q, q')$ on symbol a_{i+1}. Given a word $\alpha \in \Sigma^\omega$, the PBA $\mathcal{B}$ can be thought of as an infinite state Markov chain which gives rise to the standard σ-algebra on Q^ω defined using cylinders and the standard probability measure on Markov chains [14, 10]. We shall henceforth denote the σ-algebra as $\mathcal{F}_{\mathcal{B},\alpha}$ and the probability measure as $\mu_{\mathcal{B},\alpha}$.

A *run* of the PBA $\mathcal{B}$ is an infinite sequence $\rho \in Q^\omega$. A run ρ is *accepting* if $\rho[i] \in Q_f$ for infinitely many i. A run ρ is said to be *rejecting* if it is not accepting. The set of accepting runs and the set of rejecting runs are measurable [14]. Given a word α, the measure of the set of accepting runs is said to be the *probability of accepting* α and is henceforth denoted by $\mu^{acc}_{\mathcal{B},\alpha}$; and the measure of the set of rejecting runs is said to be the *probability of rejecting* α and is henceforth denoted by $\mu^{rej}_{\mathcal{B},\alpha}$.

Hierarchical PBA**.** Intuitively, a hierarchical PBA is a PBA such that the set of its states can be stratified into (totally) ordered levels. From a state q, for each letter a, the machine can transition with non-zero probability to at most one state in the same level as q, and all other probabilistic successors belong to a higher level.

Definition 2. Given a natural number k, a PBA $\mathcal{B} = (Q, q_s, Q, \delta)$ over an alphabet Σ is said to be a *k-level hierarchical PBA (k-*PBA*)* if there is a function $\mathsf{rk} : Q \to \{0, 1, \ldots, k\}$ such that the following holds.

> Given $j \in \{0, 1, \ldots, k\}$, let $Q_j = \{q \in Q \mid \mathsf{rk}(Q) = j\}$. For every $q \in Q$ and $a \in \Sigma$, if $j_0 = \mathsf{rk}(q)$ then $\mathsf{post}(q, a) \subseteq \cup_{j_0 \le \ell \le k} Q_\ell$ and $|\mathsf{post}(q, a) \cap Q_{j_0}| \le 1$.

The function rk is said to be a *compatible ranking function* of $\mathcal{B}$ and for $q \in Q$ the natural number $\mathsf{rk}(q)$ is said to be the *rank* or *level* of q. $\mathcal{B}$ is said to be a *hierarchical* PBA *(*HPBA*)* if $\mathcal{B}$ is k-hierarchical for some k.

Language recognized by a PBA**.** Given rational $x \in [0, 1]$ and a PBA $\mathcal{B}$ on alphabet Σ, we can define two *languages*: [4]

- $\mathcal{L}_{>x}(\mathcal{B}) = \{\alpha \in \Sigma^\omega \mid \mu^{acc}_{\mathcal{B},\,\alpha} > x\}$, and
- $\mathcal{L}_{\ge x}(\mathcal{B}) = \{\alpha \in \Sigma^\omega \mid \mu^{acc}_{\mathcal{B},\,\alpha} \ge x\}$.

The exact value of x is not important thanks to the following proposition.

Proposition 1. For any PBA (respectively, HPBA) $\mathcal{B}$, rational $x \in [0, 1)$ and rational $y \in (0, 1)$, there is a PBA (respectively, HPBA) $\mathcal{B}'$ constructible in polynomial time such that $\mathcal{L}_{>x}(\mathcal{B}) = \mathcal{L}_{>y}(\mathcal{B}')$. Furthermore, for any rational $r \in (0, 1]$ and rational $s \in (0, 1)$, there is a PBA (respectively, HPBA) $\mathcal{B}'$ constructible in polynomial time such that $\mathcal{L}_{\ge r}(\mathcal{B}) = \mathcal{L}_{\ge s}(\mathcal{B}')$.

This gives rise to the following classes of languages of infinite words.

Definition 3. Given a finite alphabet Σ, $\mathbb{L}(\mathrm{PBA}^{>0}) = \{\mathsf{L} \subseteq \Sigma^\omega \mid \exists \mathrm{PBA}\ \mathcal{B}.\ \mathsf{L} = \mathcal{L}_{>0}(\mathcal{B})\}$, $\mathbb{L}(\mathrm{PBA}^{=1}) = \{\mathsf{L} \subseteq \Sigma^\omega \mid \exists \mathrm{PBA}\ \mathcal{B}.\ \mathsf{L} = \mathcal{L}_{=1}(\mathcal{B})\}$, $\mathbb{L}(\mathrm{PBA}^{>\frac{1}{2}}) = \{\mathsf{L} \subseteq \Sigma^\omega \mid \exists \mathrm{PBA}\ \mathcal{B}.\ \mathsf{L} = \mathcal{L}_{>\frac{1}{2}}(\mathcal{B})\}$ and $\mathbb{L}(\mathrm{PBA}^{\ge\frac{1}{2}}) = \{\mathsf{L} \subseteq \Sigma^\omega \mid \exists \mathrm{PBA}\ \mathcal{B}.\ \mathsf{L} = \mathcal{L}_{\ge\frac{1}{2}}(\mathcal{B})\}$.

The classes $\mathbb{L}(\mathrm{PBA}^{>0})$ and $\mathbb{L}(\mathrm{PBA}^{=1})$ have been studied extensively in [2, 1, 9, 4]. We restrict our attention here to the classes $\mathbb{L}(\mathrm{PBA}^{>\frac{1}{2}})$ and $\mathbb{L}(\mathrm{PBA}^{\ge\frac{1}{2}})$. For hierarchical PBAs we can define classes analogous to $\mathbb{L}(\mathrm{PBA}^{>0})$, $\mathbb{L}(\mathrm{PBA}^{=1})$, $\mathbb{L}(\mathrm{PBA}^{>\frac{1}{2}})$ and $\mathbb{L}(\mathrm{PBA}^{\ge\frac{1}{2}})$; and we will call them $\mathbb{L}(\mathrm{HPBA}^{>0})$, $\mathbb{L}(\mathrm{HPBA}^{=1})$, $\mathbb{L}(\mathrm{HPBA}^{>\frac{1}{2}})$ and $\mathbb{L}(\mathrm{HPBA}^{\ge\frac{1}{2}})$ respectively.

---

[4] One does not need to explicitly consider $\mathcal{L}_{<x}(\mathcal{B})$ and $\mathcal{L}_{\le x}(\mathcal{B})$ since $\mathcal{L}_{<x}(\mathcal{B}) = \Sigma^\omega \setminus \mathcal{L}_{\ge x}(\mathcal{B})$ and $\mathcal{L}_{\le x}(\mathcal{B}) = \Sigma^\omega \setminus \mathcal{L}_{>x}(\mathcal{B})$.

Freivalds' game. Freivalds' game is a probabilistic game first presented in [7] and later used in [6] to show that checking emptiness of a PFA with non-extremal thresholds is undecidable. The game allows one to check using finite bounded memory whether two input sequences a^i and b^j, where $i, j > 0$, are of equal length.

The game on input a^i, b^j is as follows. While processing a^i, (a.1) Toss $2i$ fair coins and note if all of them turned heads. (a.2) Toss a separate set of i fair coins and note if all of them turned heads. (a.3) Toss yet another set of i fair coins and note if all of them turned heads.

While processing b^j, (b.1) Toss $2j$ fair coins and note if all of them turned heads. (b.2) Toss a separate set of j fair coins and note if all of them turned heads. (b.3) Toss yet another set of j fair coins and note if all of them turned heads.

Let A be the event that either all coins in (a.1) or (b.1) turns up heads, and B be the event that either all coins in (a.2) and (b.2) turn up heads or all coins in (a.3) and (b.3) turn up heads. The outcome of the game is said to be (a) Acc if B happens and A does not, (b) Rej if A happens and B does not, (c) $AllHeads$ if all the coins tosses (in (a.1),(a.2),(a.3),(b.1),(b.2), and (b.3)) all result in heads, and (d) $neither$ if none of the above cases hold.[5] The following observation holds about the probability of these outcomes.

Proposition 2. $Pr(Acc) \geq Pr(AllHeads)$. If $i = j$ then $Pr(Rej) = Pr(Acc)$. If $i \neq j$, $Pr(Rej) - Pr(Acc) > 3Pr(AllHeads)$.

Remark 1. In order to play the game on input a^k, b^ℓ, we need to keep track of the following pieces of information. While processing the as we need to remember 3 bits, r_1, r_2, and r_3, where r_i records whether any of the coins tossed in $(a.i)$ resulted in tails. Then while processing the bs we need to 6 bits of information — the first 3 bits to remember the results of the experiments conducted while processing the as, and the second set of 3 bits s_1, s_2, and s_3 to remember if any of the coins tossed in $(b.i)$ resulted in tails. Thus, implementing it as a finite state machine requires $2^3 + 2^6 = 72$ states. Initially, all the bits being recorded are 0, denoting that we have not seen any tails in any of the trials. Next observe that once one of these bits (say r_i) changes to 1, it will never switch back to 0. While processing the sequence of as, we will first (possibly) change the settings for the r_is and then change the s_is when processing the bs. Thus, this game can be played using a finite state machine with a hierarchical structure, where the rank of a state, records the number of r_is that are 1 and the number of s_is that are 1, giving us 8 levels.

3 Examples

Example 1. **(Recognizing non-ω-regular languages).** Several example of PBAs recognizing non-ω-regular languages with non-extremal thresholds have been constructed in literature [2, 1, 9, 4, 4]. Herein, we give yet another example, which exploits the Freivalds' game [7] described in Section 2.

[5] The original Freivalds' game only considers the outcomes Acc and Rej. However, for our purposes the outcome $AllHeads$ shall prove to be useful.

Let $\Sigma = \{\mathbf{0}, \mathbf{1}, \#\}$ and consider the language $\mathsf{L} = \{\mathbf{0}^n\mathbf{1}^n\#\alpha \mid n > 0, \alpha \in \Sigma^\omega\}$. $\mathcal{L}$ is a standard example of a non-ω-regular language. We will construct a PBA $\mathcal{B}$ such that $\mathcal{L}_{>\frac{1}{2}}(\mathcal{B}) = \mathsf{L}$.

$\mathcal{B}$ is constructed as follows. It has two special absorbing states q_a and q_r. q_a is also the only accepting state of $\mathcal{B}$. $\mathcal{B}$ proceeds as follows. When the first letter is input, $\mathcal{B}$ checks if it is $\mathbf{0}$ or not. If the letter is not $\mathbf{0}$, *i.e.*, it is either $\mathbf{1}$ or $\#$, then $\mathcal{B}$ moves to q_r with probability 1 and thus the input is rejected with probability 1. If the input is $\mathbf{0}$, then $\mathcal{B}$ starts playing the Freivalds' game in order to check if the rest of the input contains a finite sequence of $\mathbf{0}$s followed by a sequence of $\mathbf{1}$s of the same length and which is followed by $\#$. As long as $\mathcal{B}$ continues seeing input $\mathbf{0}$, $\mathcal{B}$ tosses coins according to (a.1), (a.2) and (a.3) of the Freivalds' game. If $\mathcal{B}$ encounters an input different from $\mathbf{0}$, then it proceeds as follows. If the input is $\#$ then $\mathcal{B}$ transitions to q_r with probability 1. If the input is $\mathbf{1}$, then $\mathcal{B}$ tosses coins according (b.1), (b.2) and (b.3) of the Freivalds' game as long as $\mathcal{B}$ continues seeing $\mathbf{1}$. If $\mathcal{B}$ encounters input $\mathbf{0}$ then $\mathcal{B}$ transitions to q_r with probability 1. If $\mathcal{B}$ encounters input $\#$; then the transition is defined according to result of Freivalds' game as follows.

– Freivalds' game results in event Acc: $\mathcal{B}$ transitions to q_a with probability 1.
– Freivalds' game results in event Rej: $\mathcal{B}$ transitions to q_r with probability 1.
– Freivalds' game results in event $AllHeads$: $\mathcal{B}$ transitions to q_a with probability 1.
– In all other cases, $\mathcal{B}$ transitions to q_a and q_r with probability $\frac{1}{2}$.

It is easy to see that $\mathcal{B}$ is the required PBA. Infact, observations in Remark 1 imply that $\mathcal{B}$ can be taken to be a HPBA.

Example 2. **(Multi-threaded systems and bounded context switching).** Consider a system consisting of k finite state processes. The system takes inputs and changes states. At each point, one and only one process is *active*. At each point, the system may probabilistically *context switch* making a new process *active*. Otherwise, the behavior of the system is deterministic. One may want to check that on every input, the system satisfies a property specified by a deterministic Büchi automaton with probability $\geq$ threshold value. If the system is modeled as probabilistic automata $\mathcal{A}$ and the specification by Spec, then by taking the synchronous cross-product of the automaton and specification, we can obtain a PBA $\mathcal{B}$ such that the probability of system satisfying the specification on input α is exactly the probability of $\mathcal{B}$ accepting α, thus turning the verification question into a problem of deciding universality of a PBA with a non-extremal threshold. If we bound the number of context switches, then the PBA can be taken to be a HPBA.

Remark 2. Bounding the number of context switches is a technique used to make analysis of multithreaded recursive programs tractable. Our results in [4] imply that this technique will also be useful for verification of probabilistic systems with extremal thresholds. However, our results in this paper would mean that bounding context switches might not be sufficient for non-extremal thresholds.

4 Decision problems

Given a PBA $\mathcal{B}$, the problem of checking whether $\mathcal{L}_{>0}(\mathcal{B})$ is empty (or universal) was shown to be undecidable in [1] and was later proved to be $\mathbf{\Sigma}^0_2$ complete in [4]. The prob-

	Emptiness	Universality
$\mathbb{L}(\text{PBA}^{>\frac{1}{2}})$	$\mathbf{\Sigma}_2^0$-complete	$\mathbf{\Pi}_1^1$-complete
$\mathbb{L}(\text{HPBA}^{>\frac{1}{2}})$	**co-R.E.**-complete	$\mathbf{\Pi}_1^1$-complete
$\mathbb{L}(\text{PBA}^{\geq\frac{1}{2}})$	$\mathbf{\Pi}_1^1$-complete	**co-R.E.**-complete
$\mathbb{L}(\text{HPBA}^{\geq\frac{1}{2}})$	$\mathbf{\Pi}_1^1$-complete	**co-R.E.**-complete

Fig. 1. Hardness of decision problems

lem of checking whether $\mathcal{L}_{=1}(\mathcal{B})$ is empty (or universal) was shown to be **PSPACE**-complete in [4] (the emptiness problem was shown to be in **EXPTIME** in [1]). All the above problems become decidable when we restrict $\mathcal{B}$ to be hierarchical.

Although the decidability of checking whether $\mathcal{L}_{>\frac{1}{2}}(\mathcal{B})$ is empty (or universal) has not been studied explicitly in literature, undecidability of the emptiness (and universality) problems for PFAs when the acceptance threshold is $\frac{1}{2}$ implies the undecidability of checking the emptiness (and universality) of $\mathcal{L}_{>\frac{1}{2}}(\mathcal{B})$. Similarly, checking emptiness/universality of the language $\mathcal{L}_{\geq\frac{1}{2}}(\mathcal{B})$ is also undecidable. Rather surprisingly, the undecidability result continues to hold even if $\mathcal{B}$ is hierarchical. Our results on hardness of decidability are summarized in Figure 1. We begin by establishing the lower bounds.

Lemma 1. Given a hierarchical PBA $\mathcal{B}$ on alphabet Σ, the problem of checking $\mathcal{L}_{>\frac{1}{2}}(\mathcal{B}) = \emptyset$ is **co-R.E.**-hard, checking $\mathcal{L}_{>\frac{1}{2}}(\mathcal{B}) = \Sigma^\omega$ is $\mathbf{\Pi}_1^1$-hard, checking $\mathcal{L}_{\geq\frac{1}{2}}(\mathcal{B}) = \emptyset$ is $\mathbf{\Pi}_1^1$-hard and checking $\mathcal{L}_{\geq\frac{1}{2}}(\mathcal{B}) = \Sigma^\omega$ is **co-R.E.**-hard.

Proof. We prove the **co-R.E.**-hardness of checking the emptiness of $\mathcal{L}_{>\frac{1}{2}}(\mathcal{B})$. The other lower bound proofs are obtained by modifying this construction and can be found in [5].

The hardness result will reduce the halting problem of deterministic 2-counter machines to the non-emptiness problem of HPBAs with strict acceptance thresholds. We begin by outlining the broad ideas behind the construction. Let T be deterministic 2-counter machine with control states Q and a special halting state q_h. We will also assume, without loss of generality, that each transition of T changes at most one counter and the initial counter values are 0. Recall that a configuration of such a machine is of the form (q, a^{i+1}, b^{j+1}), where $q \in Q$ is the current control state, and a^i (b^j) is the unary representation of the value stored in the first counter (second counter, respectively). The input alphabet of the HPBA $\mathcal{B}_T$ that we will construct will will consist of the set Q as well as 5 symbols- ",", "(", ")", a and b. The HPBA $\mathcal{B}_T$ will have the following property: if $\rho = \sigma_1\sigma_2\cdots\sigma_n$ is a halting computation of T then $\mathcal{B}$ will accept the word $\rho\sigma_n^\omega$ with probability $> \frac{1}{2}$; if $\rho = \sigma_1\sigma_2\cdots$ is a non-halting computation of T then $\mathcal{B}_T$ will accept ρ with probability $\frac{1}{2}$; and if $\rho \in \Sigma^\omega$ is an encoding of an invalid computation (i.e., if ρ is not of the right format or has incorrect transitions) and no prefix of ρ is a valid halting computation of T then $\mathcal{B}_T$ will accept ρ with probability $< \frac{1}{2}$. Given this property we will be able to conclude that T halts iff $\mathcal{L}_{>\frac{1}{2}}(\mathcal{B}_T)$ is non-empty, thus demonstrating the **co-R.E.**-hardness of the emptiness problem.

In order to construct a HPBA $\mathcal{B}_T$ with the above properties, $\mathcal{B}_T$ must be able to check if there is a finite prefix α of input $\rho \in \Sigma^\omega$ that encodes a valid halting computation of T. This requires checking the following properties. (1) α is of the right format, i.e., it is a sequence of tuples of the form (q, a^i, b^j). (2) The first configuration is the initial configuration. (3) Successive configurations in the sequence follow because of a valid transition of T. (4) In the last configuration, T is in the halting state q_h.

Observe that checking properties (1), (2) and (4) can be easily accomplished using only finite memory. On the other hand checking (3) requires checking that the counters are updated correctly which cannot be done deterministically using finite memory. Instead it will be checked using Freivalds game described in Section 2. This check will indeed be similar to the one used in the construction of Example 1, where it is used to check that every valid input must start with a number of $\mathbf{0}$s followed an equal number of $\mathbf{1}$s followed by a $\#$. In order to check properties (1), (2), (3), and (4) above for an input ρ, $\mathcal{B}_T$ proceeds in "phases" that are informally outlined here.

- $\mathcal{B}_T$ reads the first symbol of ρ. If this first symbol is not "(", then ρ is not of the right format and so $\mathcal{B}_T$ will move to the "reject" phase. Otherwise, $\mathcal{B}_T$ will choose (probabilistically) to do one of the following: (a) Move to "check initial" phase to check if the first few symbol encode the initial configuration; (b) Move to "check transition" phase to check if the second configuration follows from the first; (c) Move to "continue" phase to ignore the first configuration and possibly check some subsequent configuration.
- *Check initial phase:* Check if the first few symbols encode the initial configuration. If they do move to "accept" phase, and if not move to "reject" phase.
- *Continue phase:* Probabilistically choose to (a) ignore input and move to accept phase; (b) ignore input and move to reject phase; (c) ignore input and stay in continue phase; or (d) if current symbol is the beginning of a configuration (i.e., "(") then move to check transition phase to check if the next two configurations correspond to a valid transition.
- *Check Transition phase:* Check if there is a prefix of the form $(q_1, a^{i_1}, b^{j_1})(q_2, a^{i_2}, b^{j_2})$ and if the configurations encoded correspond to a valid transition by playing the Freivalds game. Also check if q_2 is a halting state. Based on these checks move (probabilistically) to accept phase or reject phase.
- *Accept phase:* Ignore the input as it has been deemed to be accepted.
- *Reject phase:* Ignore the input as it has been deemed to be rejected.

Observe that the above phases can be linearly ordered and so can be implemented using a hierarchical control structure. When we spell out the details of each phase, it will also be clear that each of the checks within a phase can be implemented within a hierarchical PBA. The probability with which different options are chosen within a phase will be set to ensure that on a prefix α of ρ the following properties hold: (a) if α is the prefix of a valid computation that has not yet reached the halting state, then the probability of reaching the accept phase is the same as the probability of reaching the reject phase, (b) if α is not a valid computation (and no prefix of alpha is a valid halting computation) then the probability of reaching the reject phase is greater than the probability of reaching the accept phase, and (c) if α is a valid halting computation then the probability of reaching the accept phase is greater than the probability of reaching the reject phase. Observe that these conditions will ensure the correctness of our reduction.

Having outlined the intuitions behind the reduction, we now give the details including the probability of the various transitions. From the initial state, $\mathcal{B}_T$ on input "(" will move to check initial phase with probability $\frac{2}{3}$, move to check transition phase with probability $\frac{1}{6}$ and move to continue phase with probability $\frac{1}{6}$. On all other inputs, $\mathcal{B}_T$ moves to the reject phase with probability 1 from the initial state.

Check initial phase. Observe that the check initial phase can be carried out by a deterministic finite state machine. If an error is discovered, the $\mathcal{B}_T$ moves to the reject phase with probability 1. On the other hand, if no error is found, then $\mathcal{B}_T$ moves to accept phase and reject phase with probability $\frac{1}{2}$.

Continue phase. The continue phase is implemented by a single state q_{cont}. On input symbol "(" (denoting the start of a configuration), $\mathcal{B}_T$ stays in continue phase with probability $\frac{1}{4}$, moves to accept phase with probability $\frac{1}{4}$, moves to reject phase with probability $\frac{1}{4}$, and moves to check transition phase with probability $\frac{1}{4}$. On all other input symbols, it moves to accept phase (and reject phase) with probability $\frac{15}{32}$, and stays in q_{cont} with probability $\frac{1}{16}$.

Accept and Reject phases. Since in these phases the input is ignored, the accept phase consists of a single absorbing state q_a, and the reject state consists of a single absorbing state q_r. The state q_a (for the accept phase) is the unique accepting state of the machine.

Check transition phase. This is the most interesting part of $\mathcal{B}_T$ that requires checking if there is a prefix of the remaining input of the form $(q_1, a^{i_1}, b^{j_1})(q_2, a^{i_2}, b^{j_2})$, where (q_1, a^{i_1}, b^{j_1}) and (q_2, a^{i_2}, b^{j_2}) are successive configurations of correct computational step of T. $\mathcal{B}_T$ must check for "formatting" errors and that q_2 is right next control state — these can be accomplished by a deterministic finite state machine. The difficulty is in checking that the counter values are correct. For this, $\mathcal{B}_T$ plays the Freivalds game (see Section 2)that checks if $i = j$ in an input a^i, a^j. So to check the correctness of counter values, $\mathcal{B}_T$ plays two Freivalds games; if i_2 (j_2) is supposed to be the increment of i_1 (j_1) then we play on a^{i_1+1}, a^{i_2} (b^{j_1+1}, b^{j_2}); if it follows by a decrement then the game is played on a^{i_1}, a^{i_2+1} (b^{j_1}, b^{j_2+1}) and if the counter values are unchanged then the game is played on a^{i_1}, a^{i_2} (b^{j_1}, b^{j_2}). The Freivalds game has 4 possible outcomes: Acc, Rej, $AllHeads$, and $neither$. After playing the two games, if both result in Acc then $\mathcal{B}_T$ moves to accept phase with probability 1, and if both result in Rej then move to reject phase with probability 1. If neither of the above cases hold then $\mathcal{B}_T$'s transitions depend on whether q_2 is the halting state. If q_2 is the halting state and if both games have outcome $AllHeads$, $\mathcal{B}_T$ moves to accept phase with probability 1. In all other cases, $\mathcal{B}_T$ moves to accept and reject phase with probability $\frac{1}{2}$.

From the construction of $\mathcal{B}_T$ it is easy to see that it is an HPBA. Furthermore, it is easy to see that if T has a halting computation $\sigma_1\sigma_2...\sigma_n$ then $\mathcal{B}_T$ will accept the word $\sigma_1\sigma_2...\sigma_n\sigma_n^\omega$ with probability $> \frac{1}{2}$. If T has a non-halting computation $\sigma_1\sigma....$ then the word $\sigma_1\sigma_2...$ is accepted with probability $= \frac{1}{2}$. Now the **co-R.E.**-hardness of emptiness checking follows from the observation below.

Claim. If $\alpha \in \Sigma^w$ does not represent a valid computation of T and no prefix of α is a valid halting computation then $\mathcal{B}_T$ accepts α with probability $< \frac{1}{2}$.

Proof of the claim: If α satisfies the premise of the above claim then one the following things must happen — (1) The initial configuration is not correct, (2) α has a prefix

$\sigma_0\sigma_1 \ldots \sigma_n u$ where σ_i is of the form $(q, a^{\ell_i}, b^{\ell'_i})$ and u is incorrectly formatted, *i.e.* either $u = w_1)w_2$ where w_1 does not contain ")" and w_1 in not contained in the set $\{(q, a^r, b^s \mid q \text{ is a control state of } T, r, s \geq 1\}$ or u has a prefix w that does not contain ")" and w itself is not contained in $Pref\{(q, a^r, b^s \mid q \text{ is a control state of } T, r, s \geq 1\}$, (3) α has a finite prefix $\sigma_0\sigma_1 \ldots \sigma_n$ where σ_i is of the form $(q, a^{\ell_i}, b^{\ell'_i})$ and one of the following happens: a) control states in two consecutive configurations σ_j, σ_{j+1} are not in accordance with the transition function of T, or b) counter values in two consecutive configurations σ_j, σ_{j+1} are not in accordance with the transition function of T, (4) $\alpha \in \Sigma^* a^\omega$ or $\alpha \in \Sigma^* b^\omega$.

We consider here the most interesting case when α has a finite prefix $\sigma_0\sigma_1 \ldots \sigma_n$ where σ_i is of the form $(q, a^{\ell_i}, b^{\ell_i})$ and the first *error* in α is that the counter values in two consecutive configurations σ_j, σ_{j+1} are not in accordance with the transition function of T. Let j_0 be the first j such that the counter values in $\sigma_{j_0}, \sigma_{j_0+1}$ are not in accordance with the transition function of T. We will assume that j_0 is > 0. The case when j_0 is 0 is similarly handled. Let σ_{j_0} be (q_1, a^{r_1}, b^{s_1}) and σ_{j_0+1} be (q_2, a^{r_2}, b^{s_2}). Consider the event $CheckBefore_{j_0}$ in which $\mathcal{B}_T$ either moves to the check initial phase or moves to the check transition phase before σ_{j_0}. Note that the probability of $\mathcal{B}_T$ accepting α given that $CheckBefore_{j_0}$ happens is exactly $\frac{1}{2}$.

Let $Check_{j_0}$ be the event that $\mathcal{B}_T$ moves to the check transition phase upon encountering σ_{j_0}, $Check_{j_0+1}$ be the event that $\mathcal{B}_T$ moves to the check transition phase on encountering σ_{j_0+1} and $CheckAfter_{j_0+1}$ be the event that $\mathcal{B}_T$ moves to the check transition phase sometime after σ_{j_0+1}. The claim follows from the following observations.

- Given the event $Check_{j_0}$ happens, the probability of $\mathcal{B}_T$ transitioning to q_r is bounded away from the probability of $\mathcal{B}_T$ transitioning to q_a by at least $\frac{2}{2^{4r_1+4s_1+4r_2+4s_2+4}}$. This follows from Proposition 2.
- Given the event $Check_{j_0+1}$ happens, the difference in probability of $\mathcal{B}_T$ transitioning to q_a and the probability of $\mathcal{B}_T$ transitioning to q_r is $\leq \frac{1}{2^{4r_2+4s_2}}$.
- This implies that given that the event $CheckBefore_{j_0}$ does not happen, the difference in the probability of $\mathcal{B}_T$ transitioning to q_r and probability of $\mathcal{B}_T$ transitioning to q_a is $> \frac{1}{4}(\frac{2}{2^{4r_1+4s_1+4s_1+4s_2+4}} - \frac{1}{2^{4r_1+4s_1+17}}\frac{1}{2^{4r_2+4s_2}} - \frac{1}{2^{4r_1}+2^{4s_1}+2^{4r_2}+2^{4s_2}+33}) > 0.$

□

Observe that since HPBAs are special PBAs, the lower bounds in Lemma 1 established for HPBAs apply also to general PBAs. In addition, for general PBA, the $\mathbf{\Sigma}^0_2$-hardness of checking emptiness of $\mathcal{L}_{>0}(\mathcal{B})$ [4] coupled with Proposition 1, establishes the $\mathbf{\Sigma}^0_2$-hardness of checking the emptiness of $\mathcal{L}_{>\frac{1}{2}}(\mathcal{B})$. The lower bounds implied by Lemma 1 and the preceding arguments in this paragraph are in fact tight. The most interesting case is the **co-R.E.** decision procedure for checking the emptiness of $\mathcal{L}_{>\frac{1}{2}}(\mathcal{B})$ for HPBAs $\mathcal{B}$, which is a consequence of the proof of the fact that for a HPBA $\mathcal{B}$, $\mathcal{L}_{>\frac{1}{2}}(\mathcal{B}) \neq \emptyset$ iff $\mathcal{L}_{>\frac{1}{2}}(\mathcal{B})$ contains an ultimately periodic word. This property is not true for general PBAs (see [1]). This property is also not true for the case $\mathcal{L}_{\geq\frac{1}{2}}(\mathcal{B})$ even if we take $\mathcal{B}$ to be hierarchical. However, we can show that $\mathcal{L}_{\geq\frac{1}{2}}$ is not universal, then its complement $\Sigma^\omega \setminus \mathcal{L}_{\geq\frac{1}{2}}(\mathcal{B})$ must contain an ultimately periodic word (even in the case $\mathcal{B}$ is not hierarchical).

Lemma 2. If $\mathcal{B}$ is a HPBA on Σ and $\mathcal{L}_{>\frac{1}{2}}(\mathcal{B}) \neq \emptyset$ then $\mathcal{L}_{>\frac{1}{2}}(\mathcal{B})$ contains an ultimately periodic word. If $\mathcal{B}$ is a PBA (not necessarily hierarchical) on Σ and $\mathcal{L}_{\geq\frac{1}{2}}(\mathcal{B}) \neq \Sigma^\omega$ then $\Sigma^\omega \setminus \mathcal{L}_{\geq\frac{1}{2}}(\mathcal{B})$ contains an ultimately periodic word.

We are ready to establish the upper bounds of the decision problems.

Theorem 1. Given a PBA $\mathcal{B}$ on alphabet Σ,

- the problem of checking whether $\mathcal{L}_{>\frac{1}{2}}(\mathcal{B}) = \emptyset$ is $\mathbf{\Sigma}^0_2$-complete. If $\mathcal{B}$ is hierarchical, the problem of checking whether $\mathcal{L}_{>\frac{1}{2}}(\mathcal{B}) = \emptyset$ is **co-R.E.**-complete.
- The problem of checking whether $\mathcal{L}_{>\frac{1}{2}}(\mathcal{B}) = \Sigma^\omega$ is $\mathbf{\Pi}^1_1$-complete. The problem continues to $\mathbf{\Pi}^1_1$-complete even if we restrict $\mathcal{B}$ to the class of hierarchical PBAs.
- The problem of checking whether $\mathcal{L}_{\geq\frac{1}{2}}(\mathcal{B}) = \emptyset$ is $\mathbf{\Pi}^1_1$-complete. The problem continues to $\mathbf{\Pi}^1_1$-complete even if we restrict $\mathcal{B}$ to the class of hierarchical PBAs.
- The problem of checking whether $\mathcal{L}_{\geq\frac{1}{2}}(\mathcal{B}) = \Sigma^\omega$ is **co-R.E.**-complete. The problem continues to be **co-R.E.**-complete even if we restrict $\mathcal{B}$ to the class of hierarchical PBAs.

5 Expressiveness

Language properties of classes $\mathbb{L}(\mathrm{PBA}^{>0})$ and $\mathbb{L}(\mathrm{PBA}^{=1})$ have been extensively studied in [1, 4, 9]. The main results established therein are the following.

- The class $\mathbb{L}(\mathrm{PBA}^{=1})$ strictly contains the class of all deterministic ω-regular languages [1, 9] and is a strict subset of all languages recognized by a deterministic Büchi automata (with possibly infinite states) [4]. Therefore, $\mathbb{L}(\mathrm{PBA}^{=1})$ is not closed under complementation [1, 4, 9].
- The class $\mathbb{L}(\mathrm{PBA}^{>0})$ strictly contains the class of all ω-regular languages [1, 9] and is the Boolean closure of the class $\mathbb{L}(\mathrm{PBA}^{=1})$ [4]. Boolean closure of a class of languages $\mathcal{C}$ is the smallest class of languages which contains $\mathcal{C}$ and is closed under finite unions, finite intersections and complementation. This implies that $\mathbb{L}(\mathrm{PBA}^{>0})$ is closed under complementation, a fact that was established in [1]. Indeed, [1] shows that the complementation is recursive.

Results of [4] immediately imply that the class $\mathbb{L}(\mathrm{PBA}^{\geq\frac{1}{2}})$ is also a subset of all languages recognized by a deterministic Büchi automata (with possibly infinite states) [4] and the containment can be shown to be strict. The classes $\mathbb{L}(\mathrm{PBA}^{\geq\frac{1}{2}})$ and $\mathbb{L}(\mathrm{PBA}^{>\frac{1}{2}})$ were also shown to contain strictly the classes $\mathbb{L}(\mathrm{PBA}^{=1})$ and $\mathbb{L}(\mathrm{PBA}^{>0})$ respectively [9]. Since $\mathbb{L}(\mathrm{PBA}^{>\frac{1}{2}})$ contains all ω-regular languages (even those not recognized by deterministic Büchi automata), $\mathbb{L}(\mathrm{PBA}^{>\frac{1}{2}})$ cannot be a subset of $\mathbb{L}(\mathrm{PBA}^{\geq\frac{1}{2}})$.

The natural question that arises is whether the class $\mathbb{L}(\mathrm{PBA}^{>\frac{1}{2}})$ is a Boolean closure of the class $\mathbb{L}(\mathrm{PBA}^{\geq\frac{1}{2}})$. Observe that Theorem 1 already implies that if $\mathbb{L}(\mathrm{PBA}^{>\frac{1}{2}})$ were to be closed under complementation, the complementation cannot be recursive. We will establish that $\mathbb{L}(\mathrm{PBA}^{>\frac{1}{2}})$ is not closed under complementation thus answering the above question in the negative. Further, we will also show that the class $\mathbb{L}(\mathrm{PBA}^{\geq\frac{1}{2}})$ is not even contained in $\mathbb{L}(\mathrm{PBA}^{>\frac{1}{2}})$. In order to establish these results, we shall need the concept to **robust** PBAs.

Robust PBAs. In the context of probabilistic automata on finite strings (PFAs), [12] introduced the notion of isolated cutpoints. A real number x is said to be an isolated cutpoint for a PFA A if there is an ϵ such that for every finite word u, the probability of A accepting u is at least ϵ away from x. We extend this notion to PBAs.

Definition 4. A PBA, $\mathcal{B}$ on Σ, is said to be *x-robust* for some $x \in (0,1)$ if there is an $\epsilon > 0$ such that for any $\alpha \in \Sigma^\omega$, $|\mu^{acc}_{\mathcal{B},\alpha} - x| > \epsilon$.

Observe first that if $\mathcal{B}$ is x-robust then $\mathcal{L}_{>x}(\mathcal{B}) = \mathcal{L}_{\geq x}(\mathcal{B})$. It was shown in [12] that the languages recognized by robust PFAs are regular languages over finite words. We had extended this result for finite probabilistic monitors (FPMs) in [3]. A FPM is a PBA in which all states except one absorbing state, called *reject state*, are final states. We will demonstrate a similar result for PBAs and show that if $\mathcal{B}$ is x-robust and $\mathcal{L}_{>x}(\mathcal{B})$ is a safety language then $\mathcal{B}$ is ω-regular. The same result also holds if complement of $\mathcal{L}_{>x}(\mathcal{B})$ is a safety language. The proof essentially follows the proof of [12] except that it depends on the assumed topological properties of $\mathcal{L}_{>x}(\mathcal{B})$.

Proposition 3. Let $\mathcal{B}$ be x-robust for some $x \in (0,1)$. If either $\mathcal{L}_{>x}(\mathcal{B})$ a safety language or $\Sigma^\omega \setminus \mathcal{L}_{>x}(\mathcal{B})$ is a safety language, then $\mathcal{L}_{>x}(\mathcal{B})$ is ω-regular.

Lemma 3. There is a language $\mathsf{L} \in \mathbb{L}(\mathrm{PBA}^{\geq \frac{1}{2}})$ such that $\mathsf{L} \notin \mathbb{L}(\mathrm{PBA}^{> \frac{1}{2}})$. Furthermore, $\mathbb{L}(\mathrm{PBA}^{> \frac{1}{2}})$ is not closed under complementation.

Proof. Let $\Sigma = \{\mathbf{0}, \mathbf{1}\}$. Let $\mathsf{num}(\mathbf{0})$ be the natural number 0 and $\mathsf{num}(\mathbf{1})$ be the natural number 1. For any word $\alpha = a_0 a_1 \ldots \in \Sigma^\omega$ let $\mathsf{bin}(\alpha)$ be the real number $\sum_{i\in\mathbb{N}, i>0} \frac{\mathsf{num}(a_i)}{2^{i+1}}$. Let $\mathsf{wrd}(\frac{1}{\sqrt{2}})$ be the unique α such that $\mathsf{bin}(\alpha) = \frac{1}{\sqrt{2}}$. We had shown in [3] that there is a FPM $\mathcal{M}$ such that $\mathcal{L}_{\geq \frac{1}{16}}(\mathcal{M}) = \{\mathsf{wrd}(\frac{1}{\sqrt{2}})\}$. Let $\mathsf{L}_0 = \{\mathsf{wrd}(\frac{1}{\sqrt{2}})\}$. L_0 is a safety language but not ω-regular.

We claim that L_0 is not in $\mathbb{L}(\mathrm{PBA}^{> \frac{1}{2}})$. We proceed by contradiction. If there is a PBA $\mathcal{B}$ such that $\mathcal{L}_{> \frac{1}{2}}(\mathcal{B}) = \mathsf{L}_0$ then $\mu^{acc}_{\mathcal{B},\mathsf{wrd}(\frac{1}{\sqrt{2}})} > \frac{1}{2}$ and for any word $\beta \neq \mathsf{wrd}(\frac{1}{\sqrt{2}})$, $\mu^{acc}_{\mathcal{B},\beta} \leq \frac{1}{2}$. Clearly $\mathcal{B}$ is $\frac{1}{2} + \frac{\mu^{acc}_{\mathcal{B},\alpha} - \frac{1}{2}}{2}$-robust. Thus, L_0 should be ω-regular by Proposition 3 which contradicts the fact that L_0 is not ω-regular.

In order to see that $\mathbb{L}(\mathrm{PBA}^{> \frac{1}{2}})$ is not closed under complementation, consider the PBA $\mathcal{B}$ obtained from $\mathcal{M}$ by taking the reject state of the FPM $\mathcal{M}$ above as the only accept state. It is easy to see that $\mathcal{L}_{> \frac{15}{16}}(\mathcal{B})$ is the language $\Sigma^\omega \setminus \mathsf{L}_0$. But the complement of $\mathcal{L}_{> \frac{15}{16}}(\mathcal{B})$ is L_0 which as already observed above is not in $\mathbb{L}(\mathrm{PBA}^{> \frac{1}{2}})$. $\square$

Remark 3. Note that the FPM $\mathcal{M}$ built in the proof above to show that $\mathbb{L}(\mathrm{PBA}^{\geq \frac{1}{2}})$ is not contained in $\mathbb{L}(\mathrm{PBA}^{> \frac{1}{2}})$ is also a HPBA. Therefore, $\mathbb{L}(\mathrm{HPBA}^{\geq \frac{1}{2}}) \not\subseteq \mathbb{L}(\mathrm{HPBA}^{> \frac{1}{2}})$ and the class $\mathbb{L}(\mathrm{HPBA}^{> \frac{1}{2}})$ is also not closed under complementation.

6 **Simple** PBAs

Unlike the case of extremal thresholds, as the results in the previous sections demonstrate, HPBAs under non-extremal thresholds lose their "regularity" and "tractability" properties. In this section we introduce a special class of HPBAs that we call *simple* PBA*s* that have many nice tractable properties even under non-extremal thresholds.

We begin by formally defining *simple* PBA*s* (SPBA). A HPBA $\mathcal{B}$ is called simple if it is a 1-level HPBA and all its accepting states are at level 0, i.e., the lowest level. Recall that in a 1-level HPBA, the level of each state is either 0 or 1. Analogous to the class $\mathbb{L}(\mathrm{HPBA}^{>\frac{1}{2}})$ and $\mathbb{L}(\mathrm{HPBA}^{\geq\frac{1}{2}})$, we can define the corresponding classes for simple PBAs, namely, $\mathbb{L}(\mathrm{SPBA}^{>\frac{1}{2}})$ and $\mathbb{L}(\mathrm{SPBA}^{\geq\frac{1}{2}})$.[6]

Theorem 2. $\mathbb{L}(\mathrm{SPBA}^{>\frac{1}{2}}) = \mathbb{L}(\mathrm{SPBA}^{\geq\frac{1}{2}}) = \mathsf{DetReg}$, where DetReg is the collection of ω-regular languages recognized by deterministic finite state Büchi automata.

Proof. Observe that every deterministic Büchi automata is a simple PBA; the language remains the same no matter what threshold (> 0) we choose and whether we interpret the threshold to be strict or non-strict. Thus, one direction of the above theorem is trivial. We now prove the other direction.

Let $\mathcal{B} = (Q, q_s, Q_f, \delta)$ be a simple PBA and let $x \in (0,1)$. For a state $q \in Q$, let $\mathsf{rk}(q) \in \{0,1\}$ denote the level of q.

We show that the language $\mathcal{L}_{\geq x}(\mathcal{B})$ is a deterministic ω-regular language by constructing a deterministic Büchi automaton $\mathcal{A}$ that accepts exactly $\mathcal{L}_{\geq x}(\mathcal{B})$. The construction is based upon the observation that for a finite input u, there is at most one level 0 state q of $\mathcal{B}$ such that $\delta_u(q_s, q) > 0$. Essentially, each state of the automaton $\mathcal{A}$ is either a pair of the form (q, y) where $y \in [x, 1]$ and q is a level 0 state of $\mathcal{B}$ (i.e., $\mathsf{rk}(q) = 0$), or is the error state $error$. $\mathcal{A}$ is constructed to satisfy the following properties. If u is a finite input and q is a level 0 state such that $\delta_u(q_s, q) = y$ and $y \geq x$, then the automaton $\mathcal{A}$ goes to state (q, y) on the input u. If there is no such state q, then $\mathcal{A}$ goes to state $error$ on the input u.

Now, we give a formal definition of $\mathcal{A}$. Let $X = \{\delta_a(q, q') : a \in \Sigma, 0 < \delta_a(q, q') < 1,\ \mathsf{rk}(q) = \mathsf{rk}(q') = 0\}$. Essentially, X is the set of non-zero probabilities less than 1, associated with transitions of $\mathcal{B}$ between level 0 states. Let $Y = \{y \geq x : y = 1 \text{ or } y = p_1 \times p_2 \times \ldots \times p_m,\ p_1, \ldots, p_m \in X\}$. The set Y is finite. To see this, let $p = \max(X)$. Note that $p < 1$. Now, let l be the maximum integer such that $p^l \geq x$. It should be easy to see that each element in Y is a product of at most l numbers from X and hence Y is bounded. Let $\mathcal{A} = (Q', (q_s, 1), F', \delta')$ be a deterministic Büchi automaton where $Q' = (Q \times Y) \cup \{error\}$, $F' = Q_f \times Y$ and δ' is as given below: $\delta' = \{((q, y), a, (q', y')) \mid a \in \Sigma,\ \mathsf{rk}(q) = \mathsf{rk}(q') = 0,\ y, y' \in Y,\ y' = y \times \delta_a(q, q')\} \cup \{((q, y), a, error) \mid a \in \Sigma,\ y \in Y$ and there is no q' such that $rk(q') = 0$ and $y \times \delta_a(q, q') \in Y\} \cup \{(error, a, error) \mid a \in \Sigma\}$. It is not difficult to see that $L(\mathcal{A}) = \mathcal{L}_{\geq x}(\mathcal{B})$. Clearly, $\mathcal{A}$ is a deterministic Büchi automaton. To show that the language $\mathcal{L}_{>x}(\mathcal{B})$ is a deterministic ω-regular language, we simply modify the above construction by defining Y to be all $y > x$ which are products of members of X. □

Theorem 3. Given a simple PBA $\mathcal{B}$ and rational $x \in (0,1)$, the following problems are all decidable in polynomial time: determining if (a) $\mathcal{L}_{>x}(\mathcal{B}) = \emptyset$, (b) $\mathcal{L}_{\geq x}(\mathcal{B}) = \emptyset$, (c) $\mathcal{L}_{>x}(\mathcal{B}) = \Sigma^\omega$, and (d) $\mathcal{L}_{\geq x}(\mathcal{B}) = \Sigma^\omega$.

[6] The construction in Proposition 1 which allows one to change thresholds does not yield simple PBAs. However, the proof of Theorem 2 allows one to switch thresholds. Theorem 3 shows that emptiness and universality are polynomial-time decidable for every threshold value.

7 Conclusions and further work

In this paper, we presented a number of expressiveness and decidability results for PBAs and HPBAs when the acceptance thresholds are non-extremal. We contrasted these results with the cases when the threshold probabilities are extremal. We also considered a subclass of HPBAs, called *simple* PBAs. We showed that the class of languages accepted by them under non-extremal threshold probabilities is exactly the class of deterministic ω-regular languages.

For an HPBA $\mathcal{B}$, checking the emptiness (and universality) of $\mathcal{L}_{>\frac{1}{2}}(\mathcal{B})$ is undecidable. In contrast the same problems are decidable if $\mathcal{B}$ is a simple PBA. Simple PBAs are a special class of 1-level HPBA. It would be interesting to see if the decidability result can be extended to all 1-level HPBAs. It will also be interesting to investigate use of simple PBAs for modeling practical systems that may fail. Investigation of other interesting subclasses of PBAs and HPBAs, for which the emptiness and universality problems are decidable for non-extremal threshold probabilities, are also interesting future work.

Acknowledgements. The authors would like to thank anonymous referees for their valuable comments. A. Prasad Sistla was supported by NSF-0720525, NSF CCF-0916438, NSF CNS-1035914 and Mahesh Viswanathan was supported by NSF CCF 0448178, NSF CCF 1016989, and NSF CNS 1016791.

References

1. C. Baier, N. Bertrand, and M. Größer. On decision problems for probabilistic Büchi automata. In *Proceedings of FoSSaCS*, pages 287–301, 2008.
2. C. Baier and M. Größer. Recognizing ω-regular languages with probabilistic automata. In *Proceedings of LICS*, pages 137–146, 2005.
3. R. Chadha, A. P. Sistla, and M. Viswanathan. On the expressiveness and complexity of randomization in finite state monitors. *J. of the ACM*, 56(5), 2009.
4. R Chadha, A. P. Sistla, and M. Viswanathan. Power of randomization in automata on infinite strings. In *Proceedings of CONCUR*, pages 229–243, 2009.
5. R Chadha, A. P. Sistla, and M. Viswanathan. Probabilistic Büchi automata with non-extremal acceptance thresholds. Technical Report LSV-10-19, LSV, ENS Cachan, France, 2010.
6. A. Condon and R. J. Lipton. On the complexity of space bounded interactive proofs (extended abstract). In *Proceedings of FOCS*, pages 462–467, 1989.
7. R. Freivalds. Probabilistic two-way machines. In *Proceedings of MFCS*, pages 33–45, 1981.
8. S. Goldwasser and S. Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *STOC*, pages 365–377, 1982.
9. M. Größer. *Reduction Methods for Probabilistic Model Checking*. PhD thesis, TU Dresden, 2008.
10. J. Kemeny and J. Snell. *Denumerable Markov Chains*. Springer-Verlag, 1976.
11. A. Paz. *Introduction to Probabilistic Automata*. Academic Press, 1971.
12. M. O. Rabin. Probabilitic automata. *Inf. and Control*, 6(3):230–245, 1963.
13. A. Salomaa. *Formal Languages*. Academic Press, 1973.
14. M. Vardi. Automatic verification of probabilistic concurrent systems. In *Proceedings of FOCS*, pages 327–338, 1985.
15. G. Varghese and N. Lynch. A tradeoff between safety and liveness for randomized coordinated attack protocols. In *Proceedings of PODC*, pages 241–250, 1992.