

On Decision Problems for Probabilistic Büchi Automata

Christel Baier¹, Nathalie Bertrand^{1,2}, and Marcus Größer¹

¹ Technische Universität Dresden, Germany

² IRISA/INRIA Rennes, France

Abstract. Probabilistic Büchi automata (PBA) are finite-state acceptors for infinite words where all choices are resolved by fixed distributions and where the accepted language is defined by the requirement that the measure of the accepting runs is positive. The main contribution of this paper is a complementation operator for PBA and a discussion on several algorithmic problems for PBA. All interesting problems, such as checking emptiness or equivalence for PBA or checking whether a finite transition system satisfies a PBA-specification, turn out to be undecidable. An important consequence of these results are several undecidability results for stochastic games with incomplete information, modelled by partially-observable Markov decision processes and ω -regular winning objectives. Furthermore, we discuss an alternative semantics for PBA where it is required that almost all runs for an accepted word are accepting, which turns out to be less powerful, but has a decidable emptiness problem.

Probabilistic ω -automata have been introduced in [BG05] as probabilistic acceptors for languages over infinite words. The central idea of these models is to resolve all choices by fixed probabilistic distributions and to define the accepted language as the set of infinite words where the probability measure of the accepting runs (according to, *e.g.*, a Büchi, Rabin or Streett acceptance condition) is positive. In the paper [BG05], we mainly concentrated on expressiveness and efficiency and showed that the class of languages that are recognizable by a probabilistic Büchi automaton (PBA) strictly subsumes the class of ω -regular languages (*i.e.*, PBA are more expressive than their nondeterministic counterparts) and agrees with the class of languages that can be accepted by a probabilistic Rabin or Streett automaton (PRA/PSA). Furthermore, there are ω -regular languages that have PBA of polynomial size, while any NBA has at least exponentially many states. Another aspect that makes probabilistic ω -automata interesting is the observation that the verification problem “given a finite Markov chain \mathcal{M} and an ω -regular language L for the undesired behaviors, check whether the undesired behaviors have zero measure in \mathcal{M} ” can be answered with a PBA-representation of L by means of a simple product-construction and graph-based methods, while the methods known for a representation of L by a standard Büchi automaton (alternating or nondeterministic) are more complex since they rely on some kind of powerset construction [Var85, CY95, BRV04].

The purpose of this paper is to study algorithmic problems for PBA in more detail and to provide answers to several questions that were left open in [BG05]. Our main results are:

- (1) a complementation operator for PBA
- (2) the fact that the language accepted by a PBA might depend on the precise transition probabilities
- (3) the undecidability of the emptiness problem for PBA, and various related problems
- (4) the decidability of the emptiness problem for PBA with a (non-standard) almost-sure semantics

To provide a complementation operator for PBA we use a technique that resembles the complementation of nondeterministic Büchi automata by means of Safra’s algorithm [Saf88] and relies on (i) a transformation of a given PBA \mathcal{P} into an equivalent PRA \mathcal{P}_R that accepts each infinite word with probability 0 or 1, (ii) the complementation of the Rabin acceptance condition in \mathcal{P}_R to obtain a PSA \mathcal{P}_S for the complement language and (iii) a transformation of \mathcal{P}_S into an equivalent PBA by means of techniques presented in [BG05].

At a first glance, the undecidability of the emptiness problem for PBA might not be astonishing given the undecidability of the emptiness problem for Rabin’s probabilistic finite automata (PFA) [Rab63, Paz71]. However, PFA are equipped with a positive threshold for the acceptance probability, while the accepted language of a PBA just requires positive acceptance probability (which is not of interest for PFA as for finite words the criterion “positive acceptance probability” agrees with the existence of an accepting run). In fact, (2) and (3) are surprising since for the verification of finite probabilistic systems (Markov chains or Markov decision processes) against ω -regular properties, the precise transition probabilities are irrelevant and a simple graph analysis suffices, as long as one is interested in qualitative questions [HSP83, Var85, CY95].

The undecidability of the emptiness problem has several important consequences. First, together with the effectiveness of complementation, it implies that all interesting algorithmic problems for PBA (such as checking emptiness, universality or equivalence) as well as all relevant verification problems for finite-state nondeterministic systems (with or without probabilism) where the desired or undesired behaviors are specified by a PBA are undecidable. Second, several undecidability results can be established for *stochastic games with incomplete information*. More precisely, we show the undecidability of the questions whether there exists an observation-based strategy such that a Büchi condition holds with positive probability or whether there is an observation-based strategy such that a coBüchi condition holds almost surely. This even holds for stochastic games with a single nondeterministic player, modelled by partially observable Markov decision processes (POMDP) which can be seen as a generalization of PBA. Although several undecidability results have been established for POMDPs and *quantitative* properties [MHC03, GD07], we are not aware of any other undecidability result for POMDPs and *qualitative* properties. Our results might be of

interest to several research communities as POMDPs are widely used in various applications like elevator controlling, autonomous robot planning, network troubleshooting or health care policymaking (see [Cas98] for a survey).

We finally discuss an alternative semantics for PBA which defines the accepted language of a PBA by the requirement that almost all runs are accepting. PBA with the *almost-sure semantics* turn out to be less powerful than PBA with the standard semantics, but checking emptiness is decidable for them. In contrast to the above undecidability results for POMDPs, we provide a decision algorithm for the almost-sure (repeated) reachability problem for POMDPs, that is, given a POMDP \mathcal{M} and a state set F , check whether there is an observation-based strategy for \mathcal{M} that ensures to visit F (infinitely often) with probability 1. This extends former results on the decidability of special cases of the qualitative model checking problem for POMDPs [dA99] where the confinement problem (which asks whether an invariant can hold with positive probability) has been addressed.

Organization of the paper Section 1 briefly recalls the basic definitions of probabilistic Büchi automata and (partially observable) Markov decision processes. The complementation of PBA is described in Section 2. Several undecidability results for PBA (as well as POMDPs) are given in Section 3. An alternative semantics for PBA is introduced and studied in Section 4. Section 5 concludes the paper.

1 Preliminaries

Throughout the paper, we assume familiarity with formal languages and non-deterministic automata over finite and infinite words, see *e.g.* [Tho90, PP04, GTW02]. We just recall the main concepts of probabilistic ω -automata with Büchi or other acceptance conditions and (partially observable) Markov decision processes. For further details we refer respectively to [BG05] and [Put94].

Probabilistic Büchi automata (PBA) can be seen as nondeterministic Büchi automata where the nondeterminism is resolved by a probabilistic choice: for any state q and letter $a \in \Sigma$ either q does not have any a -successor or there is a probability distribution for the a -successors of q . Formally, a PBA over the alphabet Σ is a tuple $\mathcal{P} = (Q, \delta, \mu, F)$ where Q is a finite state space, $\delta : Q \times \Sigma \times Q \rightarrow [0, 1]$ the transition probability function such that for all $q \in Q$ and $a \in \Sigma$,

$$\sum_{p \in Q} \delta(q, a, p) \in \{0, 1\},$$

μ the initial distribution, i.e., μ is a function $Q \rightarrow [0, 1]$ with $\sum_{q \in Q} \mu(q) = 1$, and $F \subseteq Q$ the set of accepting states. The states $q \in Q$ where $\mu(q) > 0$ are called initial. A *run* for an infinite word $w = a_1 a_2 \dots \in \Sigma^\omega$ is an infinite sequence $\pi = p_0, p_1, p_2, \dots$ of states in Q such that p_0 is initial and $p_{i+1} \in \delta(p_i, a_{i+1}) = \{q : \delta(p_i, a_{i+1}, q) > 0\}$ for all $i \geq 0$. $\text{Inf}(\pi)$ denotes the set of states that are visited infinitely often in π . Run π is called *accepting* if $\text{Inf}(\pi) \cap F \neq \emptyset$. Given an

infinite input word $w \in \Sigma^\omega$, the behavior of \mathcal{P} is given by the infinite Markov chain that is obtained by unfolding \mathcal{P} into a tree using w as a “scheduling policy”. We can therefore apply standard concepts for Markov chains (σ -algebra on the infinite paths and probability measure [KSK66, Kul95, Ste94]) to define the *acceptance probability* of w in \mathcal{P} , denoted $\text{Pr}_{\mathcal{P}}(w)$ or briefly $\text{Pr}(w)$, by the probability measure of the set of accepting runs for w in \mathcal{P} . The *accepted language* of \mathcal{P} is then defined as

$$\mathcal{L}(\mathcal{P}) = \{w \in \Sigma^\omega \mid \text{Pr}_{\mathcal{P}}(w) > 0\}.$$

The language of a PBA \mathcal{P} might be different from the language of the NBA that is obtained from \mathcal{P} by ignoring the probabilities. However, DBA and NBA that are *deterministic in limit* [Var85, CY95] can be viewed as special instances of PBA (arbitrary probabilities in $]0, 1]$ can be attached to the edges). Since each NBA can be transformed into an equivalent one that is deterministic in limit [Var85, CY95], each ω -regular language can be represented by a PBA. However, there are PBA that accept non- ω -regular languages. For example, the PBA \mathcal{P}_λ depicted in Fig. 1 with $0 < \lambda < 1$ accepts the following non- ω -regular language:

$$\mathcal{L}(\mathcal{P}_\lambda) = \{a^{k_1}ba^{k_2}ba^{k_3}b\dots \mid k_1, k_2, k_3, \dots \in \mathbb{N}_{\geq 1} \text{ s.t. } \prod_{i=1}^\infty (1 - \lambda^{k_i}) > 0\}.$$

Here and in the rest of the paper, we depict accepting states by boxes.

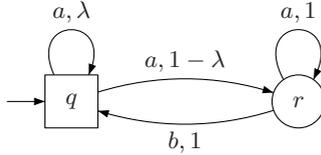


Fig. 1. PBA \mathcal{P}_λ with $0 < \lambda < 1$

Similarly, *probabilistic Rabin* and *Streett automata* (PRA and PSA, respectively) are defined as tuples $\mathcal{P} = (Q, \delta, \mu, Acc)$ where Q, δ and μ are as above, and Acc is a finite set of pairs (H, K) with $H, K \subseteq Q$. The accepted language of a PRA or PSA is defined as for PBA, but with an adapted notion of accepting runs. A run $\pi = p_0, p_1, p_2 \dots$ is accepting in a PRA if there is a pair $(H, K) \in Acc$ such that $\text{Inf}(\pi) \subseteq H$ and $\text{Inf}(\pi) \cap K \neq \emptyset$, whereas it is accepting in a PSA if for all pairs $(H, K) \in Acc$ either $\text{Inf}(\pi) \cap H = \emptyset$ or $\text{Inf}(\pi) \cap K \neq \emptyset$. Note that any PRA or PSA \mathcal{P} can be transformed into an equivalent PBA whose size is polynomially bounded in the size of \mathcal{P} [BG05].

A *Markov decision process* (MDP) is a tuple $\mathcal{M} = (Q, Act, \delta, \mu)$ where Q is a finite set a states, $\delta : Q \times Act \times Q \rightarrow [0, 1]$ a transition probability function and μ an initial distribution. The behaviour of an MDP is determined by a device, the *scheduler*, that resolves the nondeterministic choices: a scheduler for \mathcal{M} is any (history-dependent) function that selects an action for the current state i.e., a function $\mathcal{U} : S^* \rightarrow Act$ such that $\mathcal{U}(s_0 \dots s_n) = \alpha$ implies $\delta(s_n, \alpha, t) > 0$ for some state t .

A *partially observable* MDP (POMDP) is a pair (\mathcal{M}, \sim) consisting of an MDP and an equivalence relation $\sim \subseteq Q \times Q$ over the states of \mathcal{M} such that for all states $s, t \in Q$, if $s \sim t$ then the sets of actions enabled in s and t are equal. Given a POMDP (\mathcal{M}, \sim) , an observation-based scheduler \mathcal{U} is a scheduler for the underlying MDP \mathcal{M} that is consistent with \sim , *i.e.* which satisfies $\mathcal{U}(s_0s_1 \dots s_n) = \mathcal{U}(t_0t_1 \dots t_m)$ if $n = m$ and $s_i \sim t_i$ for $0 \leq i \leq m$.

Given a total PBA \mathcal{P} (a PBA that has transitions for each pair of a state and input letter) and the trivial equivalence relation $\sim = Q \times Q$, the pair (\mathcal{P}, \sim) forms a POMDP, where an observation-based scheduler represents an input word for the PBA \mathcal{P} (here $Act = \Sigma$).

2 Complementation of PBA

The question whether the class of languages recognizable by PBA is closed under complementation was left open in [BG05]. We show here that for each PBA \mathcal{P} there exists a PBA that accepts the complement of $\mathcal{L}(\mathcal{P})$. Before providing a complementation operator for PBA, we consider the PBA $\widetilde{\mathcal{P}}_\lambda$ in Fig. 2 which accepts the following language (see appendix):

$$\widetilde{L}_\lambda = \{a^{k_1}ba^{k_2}ba^{k_3}b \dots \mid k_1, k_2, k_3 \dots \in \mathbb{N}_{\geq 1} \text{ s.t. } \prod_{i=1}^\infty (1 - \lambda^{k_i}) = 0\}.$$

\widetilde{L}_λ is thus roughly the complement of the language accepted by the PBA \mathcal{P}_λ shown in Fig. 1. More precisely, it holds that $L_\lambda = (a+b)^\omega \setminus \mathcal{L}(\mathcal{P}_\lambda)$. Hence, $\widetilde{\mathcal{P}}_\lambda$ combined with a PBA for $(a+b)^*a^\omega$, $b(a+b)^\omega$ and $(a+b)^*bb(a+b)^\omega$ yields a PBA that recognizes the complement of $\mathcal{L}(\mathcal{P}_\lambda)$.

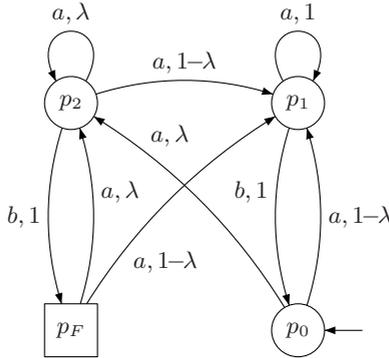


Fig. 2. PBA $\widetilde{\mathcal{P}}_\lambda$ with $0 < \lambda < 1$

Theorem 1. *For each PBA \mathcal{P} there exists a PBA \mathcal{P}' of size $\mathcal{O}(\exp(|\mathcal{P}|))$ such that $\mathcal{L}(\mathcal{P}') = \Sigma^\omega \setminus \mathcal{L}(\mathcal{P})$. Moreover, \mathcal{P}' can be effectively constructed from \mathcal{P} .*

Proof (sketch). The idea for the complementation of a given PBA \mathcal{P} is to provide the following series of transformations

$$\begin{aligned} \text{PBA } \mathcal{P} &\xrightarrow{(1)} \text{0/1-PRA } \mathcal{P}_R \text{ with } \mathcal{L}(\mathcal{P}_R) = \mathcal{L}(\mathcal{P}) \\ &\xrightarrow{(2)} \text{0/1-PSA } \mathcal{P}_S \text{ with } \mathcal{L}(\mathcal{P}_S) = \Sigma^\omega \setminus \mathcal{L}(\mathcal{P}_R) \\ &\xrightarrow{(3)} \text{PBA } \overline{\mathcal{P}} \text{ with } \mathcal{L}(\overline{\mathcal{P}}) = \mathcal{L}(\mathcal{P}_S) \end{aligned}$$

where 0/1-PRA denotes a PRA with $\Pr_{\mathcal{P}_R}(w) \in \{0, 1\}$ for each word $w \in \Sigma^\omega$. Step (2) is obvious as it relies on the duality of Rabin and Streett acceptance. In step (3), we may use the polynomial transformation from PSA to PBA presented in [BG05]. The most interesting part is step (1), which has some similarities with Safra's determinization algorithm for NBA and also relies on some kind of powerset construction. However, we argue that the probabilistic setting is slightly simpler: instead of organizing the potential accepting runs in Safra trees, we may deal with up to n independent *sample runs* (where n is the number of states in \mathcal{P}) that are representative for all potential accepting runs. The idea is to represent the current states of the sample runs by tuples $\langle p_1, \dots, p_k \rangle$ of pairwise distinct states in \mathcal{P} . Whenever two sample runs meet at some point, say the next states p'_1 and p'_2 in the first two sample runs agree, then they are merged, which requires a shift operation for the other sample runs and yields a tuple of the form $\langle p'_1, p'_3, \dots, p'_k, \dots, r, \dots \rangle$ where $p_i \rightarrow p'_i$ stands for the move in the i -th sample run. Additionally, new sample runs are generated in case the original PBA \mathcal{P} can be in an accepting state $r \notin \{p'_1, \dots, p'_k\}$. The Rabin condition serves to express the condition that at least one of the sample runs enters the set F of accepting states in \mathcal{P} infinitely often and is a proper run in \mathcal{P} (i.e., is affected by the shift operations only finitely many times). The details of this construction are complicated and omitted here. \square

3 Undecidability Results

A natural question that arises with automata is whether the accepted language of a given automaton is empty. The decidability of this problem for PBAs was open, since in [BG05] the emptiness problem was only treated for *uniform* PBA, a subclass of PBA, which are as expressive as ω -regular languages. It was shown there that checking emptiness is decidable for uniform PBAs and that the problem, given a uniform PBA \mathcal{P} , whether $\mathcal{L}(\mathcal{P}) \neq \emptyset$ is NP-hard. The decidability followed from a transformation of a uniform PBA into an equivalent NSA. For the full class of PBA we present the following result.

Theorem 2. *The emptiness problem for PBA is undecidable.*

Our proof for Theorem 2 given below relies on a reduction from a variant of the emptiness problem for PFA, using the fact that modifying the transition probabilities can affect the accepted language of a PBA. To see this last point, we consider again the PBA represented in Fig. 1 and show that:

Proposition 3. For $0 < \lambda < \frac{1}{2} < \eta < 1$, $\mathcal{L}(\mathcal{P}_\lambda) \neq \mathcal{L}(\mathcal{P}_\eta)$.

Recall that $\mathcal{L}(\mathcal{P}_\lambda) = \{a^{k_1}ba^{k_2}b \dots \mid \prod_{i \geq 1} (1 - \lambda^{k_i}) > 0\}$. Proposition 3 is an immediate consequence of the following lemma (using $n = 2$):

Lemma 4. For each $n \in \mathbb{N}_{\geq 2}$ there exists a sequence $(k_i)_{i \geq 1}$ such that

$$\prod_{i \geq 1} (1 - \lambda^{k_i}) > 0 \text{ if and only if } \lambda < \frac{1}{n}.$$

Proof. Given $n \in \mathbb{N}_{\geq 2}$, we define the sequence $(k_i)_{i \geq 1}$ in the following way: the first n elements are set to 1, then the n^2 following elements are set to 2, the n^3 next elements set to 3, etc. The sequence $(k_i)_{i \geq 1}$ is non-decreasing, and defined by plateaux of increasing values and exponentially increasing length. We show that $\prod_i (1 - \lambda^{k_i})$ is positive if and only if $\lambda < \frac{1}{n}$. To see this, we consider the series $\sum_i \log(1 - x^{k_i})$ which converges if and only if $\prod_i (1 - x^{k_i})$ is positive. Now, $\sum_i \log(1 - x^{k_i}) = \sum_i n^i \log(1 - x^i)$ by definition of the sequence (k_i) , and the latter series behaves as $-\sum_i n^i x^i$ (i.e. either both converge, or both diverge) since $\log(1 - \varepsilon) \sim_{\varepsilon \rightarrow 0} -\varepsilon$. Hence $\sum_i n^i \log(1 - x^i) < \infty$ if and only if $x < \frac{1}{n}$, and $\prod_i (1 - \lambda^{k_i}) > 0$ if and only if $\lambda < \frac{1}{n}$ which proves the claim. \square

The emptiness problem for PFA is known to be undecidable [Rab63,Paz71]. We use here a variant of this result, due to Madani et al [MHC03]:

Theorem 5 (Undecidability result for PFA, [MHC03]). *The following problem is undecidable: Given a constant $0 < \varepsilon < 1$ and a PFA that either accepts some string with probability at least $1 - \varepsilon$ or accepts all strings with probability at most ε , decide which is the case.*

To provide an undecidability proof of the emptiness problem for PBA (Theorem 2), we reduce the variant of the emptiness problem for PFA recalled in Theorem 5 to the *intersection problem for PBA* which takes as input two PBA \mathcal{P}_1 and \mathcal{P}_2 and asks whether $\mathcal{L}(\mathcal{P}_1) \cap \mathcal{L}(\mathcal{P}_2)$ is empty. As PBA are closed under intersection ([BG05]), this will complete the proof for Theorem 2.

Let \mathcal{R} be a PFA over some alphabet Σ and $0 < \varepsilon < \frac{1}{2}$ as in Theorem 5, i.e. such that either there exists some word w accepted by \mathcal{R} with probability strictly greater than $1 - \varepsilon$, or all words are accepted with probability less than ε . For $w \in \Sigma^*$, let $\text{Pr}_{\mathcal{R}}(w)$ denote the probability that the word w is accepted by \mathcal{R} . From the PFA \mathcal{R} and the constant ε we derive two PBA \mathcal{P}_1 and \mathcal{P}_2 such that

$$\mathcal{L}^{>\varepsilon}(\mathcal{R}) = \emptyset \text{ if and only if } \mathcal{L}(\mathcal{P}_1) \cap \mathcal{L}(\mathcal{P}_2) = \emptyset,$$

where $\mathcal{L}^{>\varepsilon}(\mathcal{R}) = \{w \in \Sigma^* \mid \text{Pr}_{\mathcal{R}}(w) > \varepsilon\}$. The alphabet for both \mathcal{P}_1 and \mathcal{P}_2 arise from the alphabet Σ of \mathcal{R} by adding new symbols $\#$ and $\$$, that is, \mathcal{P}_1 and \mathcal{P}_2 are PBA over the alphabet $\Sigma' = \Sigma \cup \{\#, \$\}$. The rough idea is to use the somehow complementary acceptance behaviour of the automata \mathcal{P}_λ and $\widetilde{\mathcal{P}}_\lambda$ (see Fig. 1 and 2). The automata \mathcal{P}_1 and \mathcal{P}_2 are designed to read words of the form $w_1 \# w_2 \# \dots w_{k_1} \$ w_1 \# w_2 \# \dots w_{k_2} \$ \dots$ where $w_i^j \in \Sigma^*$. Roughly speaking, \mathcal{P}_1 will mimick the automaton \mathcal{P}_λ and \mathcal{P}_2 will mimick $\widetilde{\mathcal{P}}_\lambda$, where reading a word

$w_i^j \#$ in \mathcal{P}_1 (resp. \mathcal{P}_2) corresponds to reading a single letter a in \mathcal{P}_λ (resp. $\widetilde{\mathcal{P}}_\lambda$). Recall that \mathcal{P}_λ and $\widetilde{\mathcal{P}}_\lambda$ accept infinite words of the form $a^{k_1} b a^{k_2} b \dots$ (depending on the k_i). The two $\$$ -symbols serve as a separator for \mathcal{P}_1 and \mathcal{P}_2 , just like the letter b does for \mathcal{P}_λ and $\widetilde{\mathcal{P}}_\lambda$. Thus, the number of $\#$ -symbols between the $(j - 1)$ st and the j th occurrence of $\#\#$ (and therefore the number of words w_i^j) corresponds to the value of k_j . Automaton \mathcal{P}_1 evolves from automaton \mathcal{P}_λ by replacing each of its two states q, r by a copy of the PFA \mathcal{R} . The transitions for the $\#$ -symbol will be defined, such that after reading a word $w_i^j \#$ in the copy of \mathcal{R} that corresponds to the state q (recall that this corresponds to reading a single letter a in \mathcal{P}_λ in state q) the automaton \mathcal{P}_1 is still in this copy of \mathcal{R} with probability $1 - \Pr_{\mathcal{R}}(w_i^j)$ and has moved to the other copy with probability $\Pr_{\mathcal{R}}(w_i^j)$, similar to the behaviour of automaton \mathcal{P}_λ upon reading the letter a in state q (it stays in q with probability λ and moves to r with probability $1 - \lambda$). The structure of \mathcal{P}_1 and \mathcal{P}_2 is shown in Fig. 3 and 4, respectively.

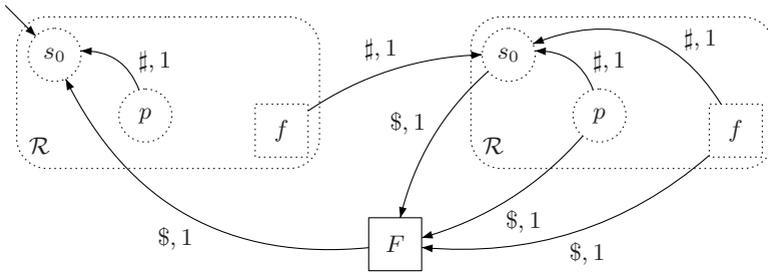


Fig. 3. PBA \mathcal{P}_1

PBA \mathcal{P}_1 is composed of two copies of the PFA \mathcal{R} (represented in dashed lines) augmented with new edges using the additional symbols $\#$ and $\$$. For simplicity, we represented only one initial and one final state from \mathcal{R} , called s_0 and f respectively. The initial states of \mathcal{P}_1 are the initial states of the first copy of \mathcal{R} . From any final state of the first copy of \mathcal{R} , the PBA \mathcal{P}_1 can reach each initial state of \mathcal{R} in the second copy (but no other state) while reading the symbol $\#$. (That means, each initial state of \mathcal{R} of the second copy is reached with the initial probability of \mathcal{R} .) Upon reading the symbol $\#$ in a non-final state p of the first copy of \mathcal{R} , the automaton \mathcal{P}_1 proceeds to each initial state of the first copy of \mathcal{R} (again the initial distribution of \mathcal{R} is assumed). Consuming the symbol $\$$ in some (final or non-final) state of the second copy, \mathcal{P}_1 enters with probability 1 the special state F , which is the unique accepting state of \mathcal{P}_1 . Reading the second $\$$ symbol, \mathcal{P}_1 moves on to an initial state.

The language of this PBA is the following:

$$\mathcal{L}(\mathcal{P}_1) = \left\{ w_1^1 \# w_2^1 \# \dots w_{k_1}^1 \$ w_1^2 \# w_2^2 \# \dots w_{k_2}^2 \$ \$ \dots \mid w_i^j \in \Sigma^* \right. \\ \left. \text{and } \prod_{j \geq 1} \left(1 - \left(\prod_{i=1}^{k_j-1} (1 - \Pr_{\mathcal{R}}(w_i^j)) \right) \right) > 0 \right\}.$$

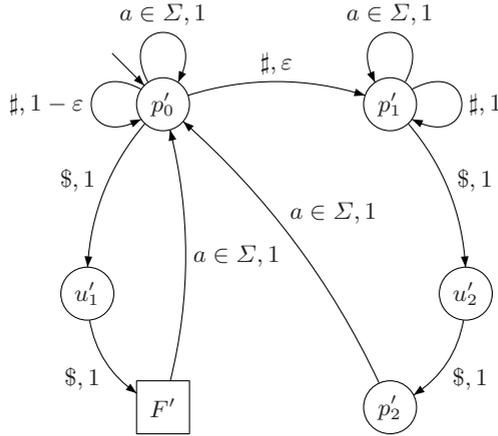


Fig. 4. PBA \mathcal{P}_2

PBA \mathcal{P}_2 (Fig. 4) does not depend on the structure of the given PFA \mathcal{R} , but only on ε and the alphabet Σ . Its accepted language is :

$$\mathcal{L}(\mathcal{P}_2) = \left\{ v_1 \$ \$ v_2 \$ \$ \dots \mid v_i \in (\Sigma \cup \{\#\})^* \text{ and } \prod_{i \geq 1} (1 - (1 - \varepsilon)^{|v_i|_{\#}}) = 0 \right\},$$

where $|v|_{\#}$ is the number of $\#$ symbols in the word $v \in (\Sigma \cup \{\#\})^*$.

Given \mathcal{P}_1 and \mathcal{P}_2 , let us now detail the correctness of the reduction, *i.e.*, prove that the language $\mathcal{L}^{>\varepsilon}(\mathcal{R}) = \{w \in \Sigma^* \mid \Pr_{\mathcal{R}}(w) > \varepsilon\}$ of \mathcal{R} for the threshold ε is empty if and only if $\mathcal{L}(\mathcal{P}_1) \cap \mathcal{L}(\mathcal{P}_2) = \emptyset$.

\Rightarrow : Assume that $\mathcal{L}^{>\varepsilon}(\mathcal{R})$ is empty, *i.e.* for all finite words $w \in \Sigma^*$ we have: $\Pr_{\mathcal{R}}(w) \leq \varepsilon$. Let $\tilde{w} \in \mathcal{L}(\mathcal{P}_2)$. The goal is to prove that $\tilde{w} \notin \mathcal{L}(\mathcal{P}_1)$. Since $\tilde{w} \in \mathcal{L}(\mathcal{P}_2)$, \tilde{w} can be written as

$$\tilde{w} = v_1 \$ \$ v_2 \$ \$ \dots \text{ with } v_i \in (\Sigma \cup \{\#\})^* \text{ and } \prod_i (1 - (1 - \varepsilon)^{|v_i|_{\#}}) = 0.$$

The subwords v_i can be decomposed according to the occurrences of the symbol $\#$. That is,

$$\tilde{w} = w_1^1 \# w_2^1 \# \dots w_{k_1}^1 \$ \$ w_1^2 \# w_2^2 \# \dots w_{k_2}^2 \$ \$ \dots \text{ with } |v_i|_{\#} = k_i - 1.$$

Hence $\tilde{w} \in \mathcal{L}(\mathcal{P}_2)$ implies $\prod_i (1 - (1 - \varepsilon)^{k_i - 1}) = 0$. However:

$$\begin{aligned} \prod_j \left(1 - \prod_{i=1}^{k_j - 1} (1 - \Pr_{\mathcal{R}}(w_i^j))\right) &\leq \prod_j \left(1 - \prod_{i=1}^{k_j - 1} (1 - \varepsilon)\right) \text{ since } \mathcal{L}^{>\varepsilon}(\mathcal{R}) = \emptyset \\ &= \prod_j \left(1 - (1 - \varepsilon)^{k_j - 1}\right) \\ &= 0 \text{ since } \tilde{w} \in \mathcal{L}(\mathcal{P}_2). \end{aligned}$$

Hence, $\tilde{w} \notin \mathcal{L}(\mathcal{P}_1)$. Since this holds for any $\tilde{w} \in \mathcal{L}(\mathcal{P}_2)$, we conclude that $\mathcal{L}(\mathcal{P}_1) \cap \mathcal{L}(\mathcal{P}_2) = \emptyset$.

\Leftarrow : Assume now that $\mathcal{L}^{>\varepsilon}(\mathcal{R}) \neq \emptyset$. By assumption on the PFA \mathcal{R} , this means that there exists a finite word $w \in \Sigma^*$ such that $\Pr_{\mathcal{R}}(w) > 1 - \varepsilon$. We define

$$\tilde{w}_{k_1, k_2, \dots} = (w\sharp)^{k_1} w\$\$ (w\sharp)^{k_2} w\$\$ \dots,$$

and prove that there exists a sequence k_1, k_2, \dots , such that $\tilde{w}_{k_1, k_2, \dots} \in \mathcal{L}(\mathcal{P}_1) \cap \mathcal{L}(\mathcal{P}_2)$. The acceptance probability of $\tilde{w}_{k_1, k_2, \dots}$ in \mathcal{P}_1 is

$$\begin{aligned} \prod_j \left(1 - \prod_{i=1}^{k_j} (1 - \Pr_{\mathcal{R}}(w))\right) &= \prod_j \left(1 - (1 - \Pr_{\mathcal{R}}(w))^{k_j}\right) \\ &> \prod_j \left(1 - (1 - (1 - \varepsilon))^{k_j}\right) \\ &= \prod_j \left(1 - \varepsilon^{k_j}\right) \end{aligned}$$

On the other hand, the word $\tilde{w}_{k_1, k_2, \dots}$ can be written as $v_1\$\$v_2\$\$ \dots$ with $v_i \in (\Sigma \cup \{\sharp\})^*$ and $|v_i|_{\sharp} = k_i$. Hence, $\prod_i (1 - (1 - \varepsilon)^{|v_i|_{\sharp}}) = \prod_i (1 - (1 - \varepsilon)^{k_i})$. We finally apply Lemma 4 (with $n = 2$) which yields the existence of a sequence $(k'_i)_{i \geq 1}$ that will ensure at the same time

$$\prod_{j \geq 1} (1 - \varepsilon^{k'_j}) > 0 \text{ and } \prod_{i \geq 1} (1 - (1 - \varepsilon)^{k'_i}) = 0.$$

Hence, $\tilde{w}_{k'_1, k'_2, \dots} \in \mathcal{L}(\mathcal{P}_1) \cap \mathcal{L}(\mathcal{P}_2)$ and $\mathcal{L}(\mathcal{P}_1) \cap \mathcal{L}(\mathcal{P}_2) \neq \emptyset$.

This completes the proof of Theorem 2.

Since complementation is effective for PBA, from the undecidability of the emptiness problem, we immediately get that many other interesting algorithmic problems for PBA are undecidable too.

Corollary 6 (Other undecidability results for PBA). *Given two PBA \mathcal{P}_1 and \mathcal{P}_2 , the following problems are undecidable.*

- universality:* $\mathcal{L}(\mathcal{P}_1) = \Sigma^\omega$?
- equivalence:* $\mathcal{L}(\mathcal{P}_1) = \mathcal{L}(\mathcal{P}_2)$?
- inclusion:* $\mathcal{L}(\mathcal{P}_1) \subseteq \mathcal{L}(\mathcal{P}_2)$?

Another immediate consequence of Theorem 2 is that the verification problem for finite transition systems \mathcal{T} and PBA-specifications is undecidable. Here we assume that the states in \mathcal{T} are labelled with sets of atomic propositions of some finite set AP and consider the traces of the paths in \mathcal{T} that arise by the projection to the labels of the states. Furthermore, we assume that the given PBA has the alphabet 2^{AP} :

Corollary 7 (Verification against PBA-specifications). *The following problems are undecidable:*

- (a) *Given a transition system \mathcal{T} and a PBA \mathcal{P} , is there a path in \mathcal{T} whose trace is in $\mathcal{L}(\mathcal{P})$?*
- (b) *Given a transition system \mathcal{T} and a PBA \mathcal{P} , do the traces of all paths in \mathcal{T} belong to $\mathcal{L}(\mathcal{P})$?*

Proof. Consider a transition system \mathcal{T} such that each infinite word over the alphabet of \mathcal{P} is a trace of \mathcal{T} . Then the emptiness problem for PBA reduces to (a) and the universality problem for PBA reduces to (b). \square

As transition systems are special instances of state-labelled Markov decision processes, the following four cases of the qualitative verification problem for finite state-labelled Markov decision processes \mathcal{M} and PBA-specifications \mathcal{P} are undecidable too. Is there a scheduler \mathcal{U} for \mathcal{M} such that

- (i) $\Pr_{\mathcal{U}}(\mathcal{L}(\mathcal{P})) > 0?$
- (ii) $\Pr_{\mathcal{U}}(\mathcal{L}(\mathcal{P})) = 1?$
- (iii) $\Pr_{\mathcal{U}}(\mathcal{L}(\mathcal{P})) < 1?$
- (iv) $\Pr_{\mathcal{U}}(\mathcal{L}(\mathcal{P})) = 0?$

Indeed, problem (a) of Corollary 7 reduces to (i) (resp. (ii)) and problem (b) reduces to (iii) (resp. (iv)) when \mathcal{T} is viewed as an MDP \mathcal{M} .

Since PBA are a special case of POMDPs our results immediately imply undecidability results for POMDPs and qualitative properties. In the literature, some undecidability results for POMDPs and quantitative properties (e.g. expected rewards, approximation of the maximal reachability problem) can be found [MHC03, GD07]. However, as far as we know, the undecidability of qualitative ω -regular properties for POMDPs is a new result. As POMDPs are $1\frac{1}{2}$ -player games, the following results also apply to the setting of stochastic multi-player games with incomplete information.

Corollary 8 (Undecidability results for POMDPs). *The following problems are undecidable:*

- (a) *Given (\mathcal{M}, \sim) a finite POMDP and F a set of states in \mathcal{M} , is there an observation-based strategy \mathcal{U} for (\mathcal{M}, \sim) such that $\Pr_{\mathcal{U}}(\Box\Diamond F) > 0?$*
- (b) *Given (\mathcal{M}, \sim) a finite POMDP and F a set of states in \mathcal{M} , is there an observation-based strategy \mathcal{U} for (\mathcal{M}, \sim) such that $\Pr_{\mathcal{U}}(\Diamond\Box F) = 1?$*

4 Almost-Sure Semantics and Decidability Results

Despite the undecidability of the emptiness problem for PBA, one way to try to recover decidability results is to consider an altered semantics for PBA. More precisely, we define the almost-sure semantics of a PBA \mathcal{P} as the set of words which generate an almost-sure set of accepting runs:

$$\mathcal{L}^=1(\mathcal{P}) = \{w \in \Sigma^\omega \mid \Pr_{\mathcal{P}}(w) = 1\}$$

Let us first observe that for probabilistic Büchi automata, the switch from the standard semantics which requires positive acceptance probability to the almost-sure semantics leads to a loss of expressiveness, and the class of probabilistic Büchi automata under the almost-sure semantics is not closed under complementation. This restricted class of languages is nevertheless not included in the ω -regular languages. Before we summarize the expressiveness results for almost-sure PBA, we fix some notation. By $\mathbb{L}(\text{PBA})$ we denote the class of PBA-definable languages, i.e. $\mathbb{L}(\text{PBA}) = \{\mathcal{L}(\mathcal{P}) \mid \mathcal{P} \text{ is a PBA}\}$. Similarly, $\mathbb{L}(\text{PBA}^=1)$

denotes the class of languages definable by a PBA with the almost-sure semantics. At last, $\mathbb{L}(\omega\text{-reg})$ denotes the class of ω -regular languages.

Theorem 9 (Expressiveness of almost-sure PBA)

- (a) $\mathbb{L}(PBA^{=1}) \subsetneq \mathbb{L}(PBA)$
- (b) $\mathbb{L}(\omega\text{-reg}) \not\subseteq \mathbb{L}(PBA^{=1})$
- (c) $\mathbb{L}(PBA^{=1}) \not\subseteq \mathbb{L}(\omega\text{-reg})$
- (d) $\mathbb{L}(PBA^{=1})$ is not closed under complementation.

Proof (sketch). The proofs for items (a) and (b) are omitted here.

- (c) The PBA $\tilde{\mathcal{P}}_\lambda$ of Fig. 2 recognizes a non- ω -regular language and enjoys the property that each word is either accepted with probability 0 or with probability 1, thus $\mathcal{L}^{=1}(\tilde{\mathcal{P}}_\lambda) = \mathcal{L}(\tilde{\mathcal{P}}_\lambda)$. This shows that $\tilde{\mathcal{P}}_\lambda$ with the almost-sure semantics accepts a non- ω -regular language.
- (d) It is evident that each DBA \mathcal{P} can be viewed as a PBA and that $\mathcal{L}(\mathcal{P}) = \mathcal{L}^{=1}(\mathcal{P})$. Consider the language $(a^*b)^\omega$. It can be recognized by a DBA and hence by a PBA with the almost-sure semantics. However, its complement $(a+b)^*a^\omega$ cannot be recognized by a PBA with the almost-sure semantics. □

Remark 10. It is worth noting that the almost-sure semantics does not lead to a loss of expressiveness if Streett or Rabin acceptance is considered. That is:

$$\mathbb{L}(\text{PSA}) = \mathbb{L}(\text{PSA}^{=1}) = \mathbb{L}(\text{PRA}^{=1}) = \mathbb{L}(\text{PRA}).$$

This follows from the duality of the Streett and Rabin acceptance conditions and the results presented in section 2 as every PRA (resp. PSA) can be transformed into an equivalent PBA [BG05].

PBA with the almost-sure semantics are less expressive but simpler to handle algorithmically. As $\mathcal{L}^{=1}(\tilde{\mathcal{P}}_\lambda) = \mathcal{L}(\tilde{\mathcal{P}}_\lambda)$, Lemma 4 implies

Proposition 11. For $0 < \lambda < \frac{1}{2} < \eta < 1$, $\mathcal{L}^{=1}(\tilde{\mathcal{P}}_\lambda) \neq \mathcal{L}^{=1}(\tilde{\mathcal{P}}_\eta)$.

Thus modifying the transition probabilities can affect the accepted language of a PBA with the almost-sure semantics. However the emptiness problem “Given a PBA, does $\mathcal{L}^{=1}(\mathcal{P}) = \emptyset$?” for PBA under the almost-sure semantics is decidable. We will show a more general result, namely the decidability of the almost-sure repeated reachability problem for POMDPs (which asks whether, for a given POMDP (\mathcal{M}, \sim) and a state set F , there exists an observation-based scheduler \mathcal{U} such that $\Pr_{\mathcal{U}}(\Box \Diamond F) = 1$).

Theorem 12. The almost-sure repeated reachability problem for POMDPs is decidable.

Proof. The proof splits into two steps. We first show (Lemma 13) that the almost-sure repeated reachability problem for POMDPs reduces to the almost-sure reachability problem for POMDPs (and vice versa) and then we proof the decidability of the latter problem (Theorem 14). □

Lemma 13. *The two following problems are reducible to each other:*

- (i) *Given a POMDP (\mathcal{M}, \sim) and a set of states F , is there an observation-based scheduler \mathcal{U} with $\Pr_{\mathcal{U}}(\Box\Diamond F) = 1$?*
- (ii) *Given a POMPD (\mathcal{M}, \sim) and a set of states F , is there an observation-based scheduler \mathcal{U} with $\Pr_{\mathcal{U}}(\Diamond F) = 1$?*

Proof. Problem (ii) reduces to (i) in a straightforward manner: given an instance for (ii) we transform it into an instance for (i) by making all F -states absorbing, i.e. by removing all outgoing edges from states in F , and adding self loops for all letters, with probability one (to these same states). We now show that problem (i) is reducible to problem (ii). Let $(\mathcal{M}, \sim), F$ be an instance for (i). We define \mathcal{M}' as follows: \mathcal{M}' consists of a copy of \mathcal{M} and some additional state f . All transitions (r, a, r') in \mathcal{M} with $r \notin F$ are left unchanged. The transitions (r, a, r') in \mathcal{M} with $r \in F$ are kept, but their probabilities are divided by 2 in \mathcal{M}' . In \mathcal{M}' , we add a self-loop with probability 1 to state f for all actions $a \in Act$. Finally, for all $r \in F$ and $a \in Act$, we add a new transition (r, a, f) with probability $\frac{1}{2}$. The transformation is depicted in figure 5.

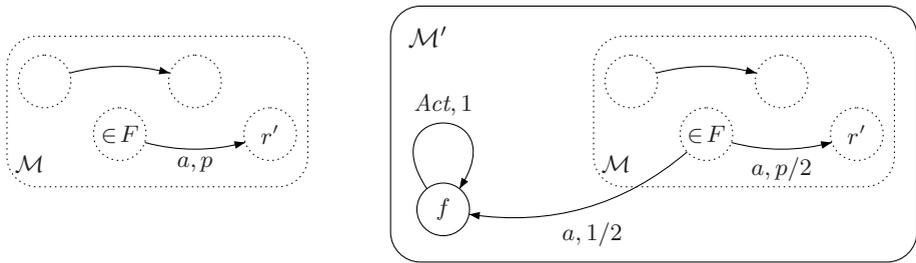


Fig. 5. Transformation from \mathcal{M} to \mathcal{M}'

The equivalence relation \sim' on $Q \cup \{f\}$ agrees with \sim on Q and $\{f\}$ forms its own equivalence class, i.e. $[s]_{\sim'} = [s]_{\sim}$ for $s \in Q$ and $[f]_{\sim'} = \{f\}$. With $F' = \{f\}$, $(\mathcal{M}', \sim'), F'$ is an instance for problem (ii) satisfying the equivalence:

$$\begin{aligned} &\exists \text{ obs.-based scheduler } \mathcal{U} \text{ s.th. } \Pr_{\mathcal{U}}^{\mathcal{M}}(\Box\Diamond F) = 1 \iff \\ &\exists \text{ obs.-based scheduler } \mathcal{U}' \text{ s.th. } \Pr_{\mathcal{U}'}^{\mathcal{M}'}(\Diamond F') = 1 \end{aligned}$$

Indeed if F is visited almost surely infinitely often in \mathcal{M} under the scheduler \mathcal{U} , F' will be almost surely visited in \mathcal{M}' under the scheduler \mathcal{U}' that mimics \mathcal{U} . Conversely, given \mathcal{U}' with $\Pr_{\mathcal{U}'}^{\mathcal{M}'}(\Diamond F') = 1$, we define \mathcal{U} to be the restriction of \mathcal{U}' on the set of path of \mathcal{M} . Then $\Pr_{\mathcal{U}}^{\mathcal{M}}(\Box\Diamond F) = 1$, since $\Pr_{\mathcal{U}'}^{\mathcal{M}'}(\Diamond\Box\neg F') > 0$ implies $\Pr_{\mathcal{U}'}^{\mathcal{M}'}(\Box\neg F') > 0$. □

Theorem 14. *The almost-sure reachability problem for POMDPs is decidable.*

Proof. We reduce the almost-sure reachability problem for POMDPs to the almost-sure reachability problem for (fully observable) MDPs, which is known

to be solvable by means of graph-algorithms. Let $\mathcal{M} = ((Q, Act, \delta, \mu), \sim)$ be a (w.l.o.g. total) POMDP and $F \subseteq Q$. We define an MDP $\mathcal{M}' = (Q', Act, \delta', \mu')$ as follows. The set of states Q' of \mathcal{M} consists of pairs (r, R) with $r \in R \subseteq [r]_{\sim}$ and an extra state q_F that has a self-loop with probability one for all $a \in Act$. Given $a \in Act$ and $R \subseteq Q$, let $R' = \delta(R \setminus F, a)$.

If $\delta(r, a) \cap F = \emptyset$ then $\delta'((r, R), a, (r', R' \cap [r']_{\sim})) = \delta(r, a, r')$ for each $r' \in Q$.

If $\delta(r, a) \cap F \neq \emptyset$ then $\delta'((r, R), a, (r', R' \cap [r']_{\sim})) = \frac{1}{2 \cdot |R \setminus F|}$ for all $r' \in R' \setminus F$ and $\delta'((r, R), a, q_F) = \frac{1}{2}$ (in case $R' \setminus F = \emptyset$, $\delta'((r, R), a, q_F) = 1$).

Moreover $\mu'(q, \{q\}) = \mu(q)$ for all $q \notin F$ and $\mu(q_F) = \sum_{r \in F} \mu(r)$. We set $F' = \{q_F\}$. This construction ensures that there exists an observation-based scheduler \mathcal{U} with $\Pr_{\mathcal{U}}^{\mathcal{M}}(\diamond F) = 1$ if and only if \mathcal{M}' has a scheduler \mathcal{U}' such that $\Pr_{\mathcal{U}'}^{\mathcal{M}'}(\diamond F') = 1$. \square

Our algorithm uses a powerset construction and hence runs in time exponential in the size of the given POMDP. However, given the EXPTIME-hardness results established by Reif [Rei84] for 2-player games with incomplete information and by de Alfaro [dA99] for POMDPs, we do not expect more efficient algorithms.

Corollary 15. *The emptiness problem for PBA with the almost-sure semantics is decidable.*

Proof. As PBA are a special case of POMDPs, this follows from Theorem 12. \square

5 Conclusion

This paper answers several open questions on probabilistic Büchi automata. We first provide a complementation operator for PBA, that somehow resembles Safra's complementation operator for NBA, but appears to be simpler as the concept of sample runs (rather than Safra trees) suffices. We then establish the undecidability of the emptiness and universality problem for PBA, which yields the undecidability of the qualitative verification problem for POMDPs against (general) ω -regular properties. Switching to an alternative *almost-sure* semantics for PBA (which turns out to be less expressive) we prove the decidability of the emptiness problem, via showing the decidability of the almost-sure repeated reachability problem and the almost-sure reachability problem for POMDPs.

References

- [BG05] Baier, C., Größer, M.: Recognizing ω -regular languages with probabilistic automata. In: Proc. 20th IEEE Symp. on Logic in Computer Science (LICS 2005), pp. 137–146. IEEE Computer Society Press, Los Alamitos (2005)
- [BRV04] Bustan, D., Rubin, S., Vardi, M.: Verifying ω -regular properties of Markov chains. In: Alur, R., Peled, D.A. (eds.) CAV 2004. LNCS, vol. 3114, pp. 189–201. Springer, Heidelberg (2004)

- [Cas98] Cassandra, A.R.: A survey of POMD applications. Presented at the AAAI Fall Symposium (1998),
<http://pomdp.org/pomdp/papers/applications.pdf>
- [CY95] Courcoubetis, C., Yannakakis, M.: The complexity of probabilistic verification. *Journal of the ACM* 42(4), 857–907 (1995)
- [dA99] de Alfaro, L.: The verification of probabilistic systems under memory-less partial-information policies is hard. In: *Proc. Workshop on Probabilistic Methods in Verification (ProbMiV 1999)*, Birmingham University, Research Report CSR-99-9, pp. 19–32 (1999)
- [GD07] Giro, S., D’Argenio, P.R.: Quantitative model checking revisited: neither decidable nor approximable. In: Raskin, J.-F., Thiagarajan, P.S. (eds.) *FORMATS 2007*. LNCS, vol. 4763, pp. 179–194. Springer, Heidelberg (2007)
- [GTW02] Grädel, E., Thomas, W., Wilke, T. (eds.): *Automata, Logics, and Infinite Games*. LNCS, vol. 2500. Springer, Heidelberg (2002)
- [HSP83] Hart, S., Sharir, M., Pnueli, A.: Termination of probabilistic concurrent programs. *ACM Transactions on Programming Languages and Systems* 5(3), 356–380 (1983)
- [KSK66] Kemeny, J.G., Snell, J.L., Knapp, A.W.: *Denumerable Markov chains*. D. Van Nostrand Co (1966)
- [Kul95] Kulkarni, V.G.: *Modeling and Analysis of Stochastic Systems*. Chapman & Hall, Boca Raton (1995)
- [MHC03] Madani, O., Hanks, S., Condon, A.: On the undecidability of probabilistic planning and related stochastic optimization problems. *Artificial Intelligence* 147(1–2), 5–34 (2003)
- [Paz71] Paz, A.: *Introduction to probabilistic automata*. Academic Press Inc., London (1971)
- [PP04] Perrin, D., Pin, J.-É.: *Infinite Words*. Pure and Applied Mathematics, vol. 141. Elsevier, Amsterdam (2004)
- [Put94] Puterman, M.: *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley and Sons, Chichester (1994)
- [Rab63] Rabin, M.O.: Probabilistic automata. *Information and Control* 6(3), 230–245 (1963)
- [Rei84] Reif, J.H.: The complexity of two-player games of incomplete information. *Journal of Computer System Sciences* 29(2), 274–301 (1984)
- [Saf88] Safra, S.: On the complexity of omega-automata. In: *Proc. 29th Symposium on Foundations of Computer Science (FOCS 1988)*, pp. 319–327. IEEE Computer Society Press, Los Alamitos (1988)
- [Ste94] Stewart, W.J.: *Introduction to the numerical solution of Markov Chains*. Princeton University Press, Princeton (1994)
- [Tho90] Thomas, W.: Automata on infinite objects. In: *Handbook of Theoretical Computer Science*, vol. B, ch. 4, pp. 133–191. Elsevier, Amsterdam (1990)
- [Var85] Vardi, M.Y.: Automatic verification of probabilistic concurrent finite-state programs. In: *Proc. 26th Symposium on Foundations of Computer Science (FOCS 1985)*, pp. 327–338. IEEE Computer Society Press, Los Alamitos (1985)