Parameter-Independent Strategies
for pMDPs via POMDPs

Sebastian Arming[1]([✉]), Ezio Bartocci[2], Krishnendu Chatterjee[3],
Joost-Pieter Katoen[4], and Ana Sokolova[1]

[1] University of Salzburg, Salzburg, Austria
sarming@cs.uni-salzburg.at
[2] TU Wien, Vienna, Austria
[3] IST Austria, Klosterneuburg, Austria
[4] RWTH Aachen University, Aachen, Germany

Abstract. Markov Decision Processes (MDPs) are a popular class of
models suitable for solving control decision problems in probabilistic
reactive systems. We consider parametric MDPs (pMDPs) that include
parameters in some of the transition probabilities to account for stochas-
tic uncertainties of the environment such as noise or input disturbances.

We study pMDPs with reachability objectives where the parameter
values are unknown and impossible to measure directly during execution,
but there is a probability distribution known over the parameter values.
We study for the first time computing parameter-independent strate-
gies that are expectation optimal, i.e., optimize the expected reachabil-
ity probability under the probability distribution over the parameters.
We present an encoding of our problem to partially observable MDPs
(POMDPs), i.e., a reduction of our problem to computing optimal strate-
gies in POMDPs.

We evaluate our method experimentally on several benchmarks: a
motivating (repeated) *learner model*; a series of benchmarks of varying
configurations of a robot moving on a grid; and a consensus protocol.

1   Introduction

Markov decision processes (MDPs) [5] are a popular class of models suitable for
solving decision making and dependability problems in a randomized environ-
ment. An MDP is a state-based model representing a probabilistic process that
satisfies the Markov property (*memorylessness*), where for each state it is pos-
sible to choose nondeterministically some action-labeled transitions governing
the probability distribution to end up in the next state. MDPs are employed in

several applications including the analysis of queueing systems [38], bird flocking [30], confidentiality [6] and robotics [32].

One of the main problems of interest for MDPs is the synthesis of an optimal policy (scheduler, strategy) choosing the sequence of actions that maximizes/minimizes the probability or the expected accumulated reward/cost to reach a target state. Model-checking tools such as PRISM [28] or Storm [20] provide a push-button technology to automate such analyses and to derive simple deterministic and memoryless schedulers.

We study here *parametric* Markov decision processes (pMDPs) [16,22], in which (some of) the transition probabilities depend on a set of parameters. This class allows to include unknown quantities in the model such as the fault rate or the input disturbances that are responsible for stochastic uncertainty. These quantities are often unavailable at the design time or impossible to measure directly at runtime. Intuitively, a pMDP represents a family of MDPs—one for each possible valuation of the parameters.

In the past years, there has been a great effort to solve reachability analysis in pMDPs using symbolic approaches [2,16,17,22,35,36]. These methods generally partition the parameter space in regions, associating each region to the optimal memoryless scheduler that maximizes/minimizes the probability to reach the target state. The common assumption of all these approaches is the possibility to observe the unknown quantities at some point and then to choose accordingly the best scheduler. However, this is not always feasible.

Our Contribution. We analyze pMDPs with reachability objectives[1] without assuming that parameter values are accessible directly during execution. Specifically, *we find parameter-independent strategies that are expectation ε-optimal* [2], *for $\varepsilon \geq 0$, i.e., optimize the expected reachability probability given a probability distribution over the parameters.* To achieve this goal, we consider partially observable Markov decision processes (POMDPs) where an agent cannot directly observe the environment's state and must take decisions according to its belief on the current state; the belief can be updated by interacting with the environment.

We provide an encoding of a pMDP as a POMDP where the states consist of pairs of the original pMDP states and parameter values and transitions can occur only between states with the same parameter values. We prove that solving the induced POMDP corresponds to finding parameter-independent expectation ε-optimal policies for the pMDP. Note that here memoryless policies are not sufficient for optimality, see the discussion on the motivating learner model in Sect. 5. We leverage algorithms such as *point-based value iteration* (PBVI) [34] and *Incremental Pruning* (IP) [10] to find the solution. We have implemented our approach using Storm [20] and AI-Toolbox [7].

Finally, we evaluate our approach experimentally on several benchmarks: a motivating (repeated) *learner model*; a series of benchmarks of varying configurations of a robot moving on a grid; and a consensus protocol model.

---

[1] We can deal with objectives beyond reachability as long as they are induced by a reward structure (for applicability of the available tools), see Sects. 3 and 4.

Paper Organization. Sect. 2 discusses related work. In Sect. 3 we introduce MDPs, pMDPs, and POMDPs. Section 4 presents the encoding of a pMDP in POMDP and the reduction result. In Sect. 5 we illustrate our approach on several case studies and report on experimental results, while in Sect. 6 we wrap up with conclusions and discussion of future work.

2   Related Work

Parametric probabilistic models [18,29] are a special class of Markov models where some of the transition probabilities (or rates) depend on one or more parameters that are not known a-priori. These models are particularly useful to study systems characterized by stochastic uncertainty due to the impossibility to access certain quantities (e.g., fault rates, packet loss ratios, etc.).

In the last decade, there was a great effort to study the problem of symbolic model checking of parametric probabilistic Markov chains [8,18,19,21,25,33,36]. In [18], Daws introduced a method to express the probability to reach the target state as a multivariate rational function with the domain in the parameter space. This approach was then efficiently implemented in the PARAM1 and PARAM2 tools [21] and included later on in the PRISM model checker [28].

The *parameter synthesis* problem consists of (exploiting the generated multivariate rational function and) finding the parameter values that would maximize or minimize the probability to reach the target state.

Repairing a probabilistic model [8] consists instead of solving a constrained nonlinear optimization problem where the objective function represents the minimal change in the transition probabilities such that the probability to reach the target state is constrained to a given bound.

The price to pay for these techniques is the increasing complexity of the multivariate rational functions in the presence of large models [27,29], causing the parameter synthesis to be also very computationally expensive. However, the introduction of new efficient heuristics [19,25,33,36] has helped to alleviate this problem by supporting the parameter synthesis for quite large models.

This symbolic approach to parameter synthesis has been recently extended to handle also the nondeterministic choice in parametric Markov decision processes (pMDPs) [2,16,17,22,35,36], where each different sequence of inputs can induce a distinct Markov chain, resulting potentially in several multivariate rational functions.

The parameter synthesis problem for pMDP consists in solving a nonlinear program (NLP) with multiple objectives. Recently, the authors in [17] have shown that many NLPs related to pMDP belong to a certain class of nonconvex optimization problems called signomial programs (SGPs). In the same paper, they have also introduced an approach to relax nonconvex constraints in SGPs generating geometric programs, a particular class of convex programs that can be solved in a number of steps that is polynomial in the number of variables.

Another approach proposed in [16] is based on sampling techniques (i.e., Metropolis-Hastings algorithm, particle swarm optimization, and cross-entropy method) that are used to search the parameter space. These heuristics usually do not guarantee that global optimal parameters will be found. Furthermore, when the regions of the parameters satisfying a requirement are very small, a large number of simulations is required.

All the proposed methods provide a map that relates the regions of the parameter space to the optimal memoryless scheduler that maximizes/minimizes the probability to reach the target state. The underlying assumption for these approaches is the possibility to measure the parameters during system execution. Once the values of the parameters have been measured, one can use this map to choose the best policy. In this paper, we consider a different assumption with respect to the state of the art: *We want to synthesize an expectation ε-optimal scheduler for pMDP that is independent from the possibility to measure the parameters.* To achieve this goal we show how to recast the problem into finding an ε-optimal policy for a partially observable Markov decision process (POMDP) after providing a suitable encoding. While we are not aware of any other work that establishes such a correspondence, it is worth mentioning that instead parameter synthesis for parametric Markov chains has been recently employed to find permissive finite-state controllers for POMDPs in [26].

The qualitative analysis of POMDPs has been widely studied: complexity results have been established [4,11] and symbolic algorithms [13]. However, for the general quantitative problem and its approximation the computational questions are undecidable [12,31]. Despite the undecidability there are several practical approaches such as point-based methods [39].

3   The Models

In this section we introduce the models of importance for this paper: MDPs, parametric MDPs, and partially observable MDPs. The models can be arbitrarily large, we do not impose restrictions on the size for the theoretical part of the paper.

3.1   Markov Decision Processes – MDPs

A (discrete) probability distribution on a set S is a function $\mu \colon S \to [0,1]$ with the property $\sum_{s \in S} \mu(s) = 1$. By $\mathcal{D}\,S$ we denote the set of all (discrete) probability distributions on S. For $s \in S$, we write δ_s for the Dirac distribution that assigns 1 to s.

Definition 3.1 (Markov Decision Process – MDP). *A Markov Decision Process (MDP) is a tuple $M = (S, A, T, i)$ where:*

 – *S is a set of states,*
 – *A is a set of actions,*
 – *$T \colon S \times A \to \mathcal{D}\,S$ is the transition function, and*
 – *$i \in \mathcal{D}\,S$ is the initial state distribution.*   ◇

From a given state with a given label, an MDP makes a step to a probability distribution over states that describes the probability of reaching a next state. As usual, we write $s \xrightarrow{a} \mu$ for $\mu = T(s, a)$. We will write $s \xrightarrow{a,p} t$ for $s \xrightarrow{a} \mu$ and $p = \mu(t)$, as well as $s \xrightarrow{a} t$ for $s \xrightarrow{a,p} t$ with $p > 0$. It is also common to write $T(t|s, a)$ for $T(s, a)(t)$.

Remark 3.1. In our definition no action is disabled in any state. This is somewhat unusual for MDPs, but very common for partially observable MDPs which we are interested in.

A *run* (also called path or play) of an MDP is an infinite sequence $s_0, a_0, s_1, a_1, \ldots$ in $(S \times A)^\omega$ of states and actions such that $s_i \xrightarrow{a_i} s_{i+1}$ for all $i \geq 0$. When convenient, we will also write $s_0 \cdot a_0 \cdot s_1 \cdot a_1 \ldots$ for the run $s_0, a_0, s_1, a_1, \ldots$. A *history* h is a finite prefix of a run in $(S \times A)^* \times S$. We write first(h) and last(h) for the first and last state in a history h, respectively. The *cone* Cone(h) of a history h is the set of all runs with prefix h. We write Runs for the set of all runs, Hist for the sets of all histories, and Cones for the smallest σ-algebra containing the cones of all histories.

Definition 3.2 (MDP policy). *A policy (strategy, scheduler) π for an MDP M is a map*

$$\pi \colon \text{Hist} \to \mathcal{D} A$$

from histories to probability distributions over the actions. It is a deterministic policy if the image of π consists only of Dirac distributions. It is a memoryless (or Markov*) policy if $\pi(w \cdot s) = \pi(s)$ for $w \in (S \times A)^*$ and $s \in S$.* ◇

A policy π together with the initial state distribution $i \in \mathcal{D} S$ induces a probability space (Runs, Cones, $\mathbb{P}_{\pi,i}$), i.e., a probability measure $\mathbb{P}_{\pi,i}$ on the measurable space (Runs, Cones). This construction is done in several steps: First, for a given state s, we consider a function $\mathrm{P}_{\pi,s}$ assigning a number in $[0, 1]$ to cones of histories in Hist. It is defined inductively as follows. We set $\mathrm{P}_{\pi,s}(\text{Cone}(h)) = 1$ if $h = s$ and $\mathrm{P}_{\pi,s}(\text{Cone}(h)) = 0$ if $h = t \neq s$. For $h = w \cdot a \cdot t$ we set $\mathrm{P}_{\pi,s}(\text{Cone}(h)) = \mathrm{P}_{\pi,s}(\text{Cone}(w)) \cdot \pi(w)(a) \cdot T(\text{last}(w), a)(t)$. By Carathéodory's extension theorem, the function $\mathrm{P}_{\pi,s}$ extends to a unique measure on the measurable space (Runs, Cones) which we denote by $\mathbb{P}_{\pi,s}$. Finally, given the initial state distribution $i \in \mathcal{D} S$, $\mathbb{P}_{\pi,i}$ is the measure on (Runs, Cones) defined as $\mathbb{P}_{\pi,i} = \sum_{s \in S} i(s) \cdot \mathbb{P}_{\pi,s}$.

We write $\mathbb{E}_{\pi,i}$ for the expectation operator of $\mathbb{P}_{\pi,i}$. Recall that the expectation operator of a measure μ on a measurable space (X, Σ) is defined as $E_\mu(f) = \int f d\mu$ for a measurable function $f \colon X \to \mathbb{R}$ where we consider the Borel σ-algebra on the reals. Hence $\mathbb{E}_{\pi,i}(f) = \int f d \mathbb{P}_{\pi,i}$ for a measurable function $f \colon$ Runs $\to \mathbb{R}$. We may sometimes decorate the notation of the measures and the expectation operators by superscript M, to emphasize the involved model.

We can now specify what it means to solve an MDP.

Definition 3.3 (Objective, value, solution of MDP). *Given an MDP*
$M = (S, A, T, i)$, *a* Borel objective, *also called return, is a measurable func-
tion* r: Runs $\rightarrow \mathbb{R}$. *The* value *of the MDP M for the objective r is defined as*
$\mathrm{Val}(r) = \sup_{\pi} \mathbb{E}_{\pi,i}(r)$. *A* solution *to an MDP M regarding the objective r is a
policy* π *with* $\mathbb{E}_{\pi,i}(r) = \mathrm{Val}(r)$. *A policy* π *is an* ε-solution, *for* $\varepsilon > 0$, *of M with
respect to r if* $\mathbb{E}_{\pi,i}(r)$ *is* ε-*close to* $\mathrm{Val}(r)$.   ◇

Note that a solution to an MDP need not exist. We will say objective for
a Borel objective. Some objectives arise via the *payoff* or *accumulated reward*
of runs. For MDPs with such reward-based objectives, a solution always exists.
Solving (partially observable) MDPs often refers to solving reward-based objec-
tives.

Definition 3.4 (MDP with rewards). *An MDP with rewards is a tuple*
$M_R = (M, R)$ *where* $M = (S, A, T, i)$ *is an MDP and* R: $S \times A \times S \rightarrow \mathbb{R}$ *is
the reward function.*   ◇

Upon performing a transition, an MDP with rewards collects reward as
described by the reward function. We will sometimes write $s \overset{a,p,r}{\rightsquigarrow} t$ for $s \overset{a,p}{\rightsquigarrow} t$ and
$R(s, a, t) = r$, and $s \overset{a,r}{\rightarrow} t$ for $s \overset{a}{\rightarrow} t$ and $R(s, a, t) = r$. If clear from the context,
we will drop the subscript R in an MDP M_R with rewards.

Reward structures may induce objectives as follows: The (undiscounted)
accumulated reward of a run in an MDP with rewards is

$$r_R(s_0, a_0, s_1, a_1 \dots) = \sum_{i \geq 0} R(s_i, a_i, s_{i+1}). \tag{1}$$

The accumulated reward induces the *reward objective* r_R if the above assignment
defines a measurable function r_R: Runs $\rightarrow \mathbb{R}$.[2]

Reachability Objectives
Of special interest to us is optimizing *reachability*, that is optimizing the prob-
ability to reach a target state (or a set of target states). Computing extremal
(i.e., maximal/minimal) reachability probabilities is at the heart of MDP model
checking: PCTL and LTL model checking boil down to computing reachabil-
ity probabilities. The same holds for omega-regular properties: determining the
maximal probability of any omega-regular property φ in an MDP M amounts
to determining the maximal probability to reach an accepting end component
in the product of M with a deterministic omega-automaton for φ. Verifying
PCTL properties under fair policies, i.e., policies that can almost surely reach
a state satisfying some fairness constraint, can also be reduced to computing
reachability probabilities.

---

[2] The discounted accumulated reward objective is defined in a similar way, by adding
a factor γ^i to the i-th summand in (1) with $\gamma \in [0, 1)$ being the discount factor. For
solving reachability objectives, undiscounted rewards are sufficient.

Definition 3.5 (Reachability objective). *Let $M = (S, A, T, i)$ be an MDP and $t \in T$ a state. The* reachability objective r_t *of reaching the state t is given by the indicator function of the set Runs_t of runs that reach t, i.e.,*

$$\mathrm{Runs}_t = \{s_0, a_0, s_1, a_1, \cdots \in \mathrm{Runs} \mid \exists i \geq 0.s_i = t\}$$

and $r_t(\rho) = 1$ if $\rho \in \mathrm{Runs}_t$ and $r_t(\rho) = 0$ otherwise. ◇

Note that Runs_t is a measurable set, i.e., $\mathrm{Runs}_t \in \mathrm{Cones}$ and hence r_t is a measurable function. Moreover, a solution to the reachability objective r_t is a policy π that maximizes the reachability probability, as $\mathbb{E}_{\pi,i}(r_t) = \mathbb{P}_{\pi,i}(\mathrm{Runs}_t)$.

Reachability objectives are induced by reward structures as follows. Given an MDP $M = (S, A, T, i)$ and a target state $t \in S$ we construct the MDP with rewards $M_t = (S, A, T_t, i, R_t)$ where

$$T_t(s, a) = \begin{cases} \delta_t & \text{if } s = t \\ T(s, a) & \text{otherwise} \end{cases} \qquad R_t(s, a, s') = \begin{cases} 1 & \text{if } s \neq t \text{ and } s' = t \\ 0 & \text{otherwise .} \end{cases}$$

The following standard property relates the accumulated reward of M_t to solving the reachability objective and is not difficult to show. (The condition $i(t) = 0$ is technical: in M_t the history t does not accumulate any reward.)

Proposition 3.1. *For any policy π for an MDP $M = (S, A, T, i)$ with $t \in S$ and $i(t) = 0$, the probability to reach t in M under π is the (undiscounted) accumulated reward of π in M_t, i.e., $\mathbb{P}^M_{\pi,i}(\mathrm{Runs}_t) = \mathbb{E}^{M_t}_{\pi,i}(r_{R_t})$.* □

The finite-horizon reachability objective is reachability within k steps for $k \in \mathbb{N}$ being called *horizon*. This objective is induced by the rewards via

$$r^k_{R_t}(s_0, a_0, s_1, a_1 \dots) = \sum_{0 \leq i \leq k-1} R_t(s_i, a_i, s_{i+1}) \tag{2}$$

with R_t being the reachability reward structure in M_t.

3.2   Parametric MDPs – pMDPs

Definition 3.6 (Parametric MDP – pMDP[3]). *A parametric Markov Decision Process (pMDP) is a tuple $M = (S, A, X, T, i)$ where S, A, and i are as in the definition of an MDP and*

– *X is the parameter space,*
– *$T: S \times A \to (\mathcal{D}\, S)^X$ is the transition function.* ◇

---

[3] We use the abbreviation pMDP rather than PMDP as it is common in the recent literature, see (e.g. [17,35] and as it reminds of the parameter p.).

For a pMDP M as above and a parameter point $x \in X$ we write $M(x)$ for the *evaluation of M at x*, that is the MDP $M(x) = (S, A, T_x, i)$ with $T_x(s, a) = T(s, a)(x)$.

We are interested in finding ε-optimal policies independent of the parameter, i.e., policies that are somehow ε-optimal over the whole parameter space X. Since for pMDPs the expected return is a function in the parameter, it is not a priori clear what optimality criterion to choose — see [2] for several alternatives. Here we consider the case where a parameter distribution $p \in \mathcal{D}X$ is given and we optimize the expected reward given p (this setting is called *expectation optimal* in [2])—as formalized below.

Runs, histories, and policies are defined similarly as for MDPs including in addition the parameter value, and the probability measure now also depends on p. For a parameter space X, the sample space is $\text{Runs}_X = \{(x, \rho) \mid x \in X, \rho \in \text{Runs}(M(x))\}$; the set of histories is $\text{Hist}_X = \{(x, h) \mid x \in X, h \in \text{Hist}(M(x))\}$; and the σ-algebra is the smallest σ-algebra Cones_X that contains the sets

$$\{\{x\} \times \text{Cone}(h) \mid x \in X, h \in \text{Hist}(M(x))\}.$$

Definition 3.7 (pMDP policy). *A policy π_X of a pMDP M is a map $\pi_X \colon \text{Hist}_X \to \mathcal{D}A$ that is* independent from the parameters, *i.e., that satisfies the property*

$$\pi_X(x, h) = \pi_X(y, h) \quad \textit{for all } x, y \in X, h \in \text{Hist}(M(x)) \cap \text{Hist}(M(y)). \tag{3}$$

Note that in general $\text{Hist}(M(x))$ may differ from $\text{Hist}(M(y))$ for $x \neq y$, as different parameter values may make the transition probabilities of certain transitions equal to zero. Furthermore, note that a pMDP policy, due to the requirement (3) can equivalently be defined as a map $\pi_X \colon \bigcup_{x \in X} \text{Hist}(M(x)) \to \mathcal{D}A$.

A pMDP policy π_X, induces a family of MDP policies $(\pi_x \mid x \in X)$ with π_x a policy for $M(x)$ by projection, i.e., $\pi_x(h) = \pi_X(x, h)$ for all $h \in \text{Hist}(M(x))$.

The measurable space $(\text{Runs}_X, \text{Cones}_X)$ is (isomorphic to) the disjoint union (coproduct) measurable space

$$\text{Runs}_X = \coprod_{x \in X} \text{Runs}(M(x)), \quad \text{Cones}_X = \{ \coprod_{x \in X} A_x \mid A_x \in \text{Cones}(M(x))\}$$

and by $\mathbb{P}_{\pi_X, i, p}$ we denote the measure that is the p-convex combination of the measures $\mathbb{P}_{\pi_x, i}^{M(x)}$, i.e.,

$$\mathbb{P}_{\pi_X, i, p}(\coprod_{x \in X} A_x) = \sum_{x \in X} p(x) \cdot \mathbb{P}_{\pi_x, i}^{M(x)}(A_x). \tag{4}$$

Remark 3.2. Note that $\mathbb{P}_{\pi_X, i, p}$ is the unique extension of the assignment

$$\text{P}_{\pi_X, i, p}(\{x\} \times \text{Cone}(h)) = p(x) \cdot \mathbb{P}_{\pi_x, i}^{M(x)}(\text{Cone}(h))$$

to the measurable space $(\text{Runs}_X, \text{Cones}_X)$.

We write $\mathbb{E}_{\pi_X,i,p}$ for the expectation operator of $\mathbb{P}_{\pi_X,i,p}$. The *value* of a pMDP M given parameter distribution p and objective r is $\mathrm{Val}(p,r) = \sup_{\pi_X} \mathbb{E}_{\pi_X,i,p}(r)$ where the supremum is taken over all pMDP policies π_X.

Definition 3.8 (Expectation ε–optimal policy). *A policy π_X is* expectation ε-optimal *for a pMDP M iff $\mathbb{E}_{\pi_X,i,p}(r)$ is ε-close to $\mathrm{Val}(p,r)$.*   ◇

3.3 Partially Observable MDPs – POMDPs

Definition 3.9 (Partially observable MDP – POMDP). *A* partially observable MDP *(POMDP) is a tuple $M = (S, A, T, i, \Omega, O)$ where:*

- *(S, A, T, i) is the underlying MDP,*
- *Ω is the set of observations,*
- *$O\colon S \to \Omega$ is the observation function.*   ◇

Note that our observation function is deterministic and only state-dependent, which is not a restriction [14]. Runs and histories of a POMDP are the runs and the histories of its underlying MDP. The reward structure for a POMDP is a reward structure of its underlying MDP. The observation function O extends naturally to runs and histories as follows, by slight abuse of the notation we denote all these functions by O. We have $O\colon \mathrm{Runs} \to (\Omega \times A)^\omega$ given by

$$O(s_0, a_0, s_1, a_1, \dots) \;=\; O(s_0), a_0, O(s_1), a_1, \dots$$

and similarly we define $O\colon \mathrm{Hist} \to (\Omega \times A)^* \times \Omega$.

The crucial difference to MDPs is that the policy of a POMDP can only observe the observations but not the states directly:

Definition 3.10 (POMDP policy). *A policy π for a POMDP M is a policy for the underlying MDP of M with the additional requirement that $\pi(h) = \pi(h')$ whenever $O(h) = O(h')$, for all $h, h' \in \mathrm{Hist}$.*   ◇

POMDP policies, together with the initial state distribution, also induce a probability measure over runs. The measurable space is again (Runs, Cones). The measure $\mathbb{P}_{\pi,i}$ is defined in exactly the same way as for the underlying MDP, and $\mathbb{E}_{\pi,i}$ again denotes the expectation operator. The value of a POMDP M on an objective r, also denoted by $\mathrm{Val}(r)$, is defined as $\mathrm{Val}(r) = \sup_\pi \mathbb{E}_{\pi,i}(r)$ where the supremum is taken over all *POMDP policies* π. A POMDP policy π is a *solution* of the POMDP M for an objective r, iff $\mathbb{E}_{\pi,i}(r) = \mathrm{Val}(r)$; it is an ε-*solution* for $\varepsilon > 0$ iff $\mathbb{E}_{\pi,i}(r)$ is ε-close to $\mathrm{Val}(r)$.

Finite-horizon accumulated reward objectives as defined in (2) are by far the most studied class of POMDP objectives; one might even say that solving such objectives is *the* POMDP problem [10,34].

Remark 3.3. We note two important facts for solutions of POMDPs:

(1) For POMDPs, deterministic policies are not a restriction (they are as powerful as randomized policies, but can require more memory) for any Borel objective, see [15, Lemma 1,Theorem 7].

(2) For POMDPs with reachability objectives, for ε-approximation with $\varepsilon > 0$, finite-memory policies are sufficient for optimality. This is because given any $\varepsilon > 0$, there exists a finite horizon N_ε, such that reachability within N_ε steps ε-approximates the optimal reachability probability, and for finite-horizon reachability optimal finite-memory policies are sufficient.

4   The Encoding

In this section we reduce the problem of finding an expectation-optimal policy for a pMDP to the problem of solving a POMDP, by presenting an encoding of pMDPs to POMDPs that will relate the policies in the desired way.

The main technical observation of our paper, the observation that enables the method of finding parameter-independent optimal policies for pMDPs via solving the induced POMDP, is the encoding and the correspondence result, Theorem 4.2 below. We start with presenting the encoding.

Definition 4.1 (Induced POMDP). *Given a pMDP $M = (S, A, X, T, i)$, its induced POMDP is $M' = (S \times X, A, T', S, O)$ with:*

$$T'((s, x), a)(s', x') = T(s, a)(x)(s') \cdot \delta_x(x')$$

and $O((s, x)) = s$. ◇

Hence, the encoding, i.e., the induced POMDP of a pMDP M is a POMDP with much larger state space: new states are pairs of states of M ("old" states) and parameter values in X. Transitions are only possible among new states with the same parameter value, i.e., transitions can not change the parameter values, and the transitions are inherited from the pMDP. Observations are the old states, i.e., in a new state (s, x) we can observe the old state s but not the parameter value x.

Our correspondence result is a consequence of the classical "change of variable" result of measure theory, which we recall next.

Theorem 4.1 ([24, Theorem VIII.C]). *Let (X, Σ) and (X', Σ') be measurable spaces, $f \colon (X, \Sigma) \to (X', \Sigma')$ a measurable function, $\mu \colon \Sigma \to \mathbb{R}^+$ a measure, and $\varphi' \colon X' \to \mathbb{R}^+ \cup \{\infty\}$ a measurable function.*

Let $\mu' \colon X' \to \mathbb{R}^+$ be the push-forward measure of μ along f, i.e. $\mu' = \mu \circ f^{-1}$ and let $\varphi = \varphi' \circ f \colon X \to \mathbb{R}^+ \cup \{\infty\}$.

Then

$$\int_X \varphi \, d\mu = \int_{X'} \varphi' \, d\mu',$$

that is, if one of the integrals exists, the other does too and they are equal. □

At this point, let us denote by Runs$'$, Hist$'$ and Cones$'$ the runs, histories, and the σ-algebra generated by the cones of histories of the encoding M'. Note that a policy of M' is a map $\pi' \colon \text{Hist}' \to \mathcal{D}\,A$. Moreover, note that

$$\text{Runs}' = \{(s_0, x), a_0, (s_1, x), a_1, \cdots \mid x \in X \text{ and } s_0, a_0, s_1, a_0, \cdots \in \text{Runs}(M(x))\}$$

and analogously for histories. This is a consequence of the construction of the encoding, i.e., of the fact that every run of M' involves a single parameter point as $T'((s, x), a)(s', x') = 0$ whenever $x \neq x'$.

We now state several direct consequences of the definitions needed for the correspondence result. The proofs of all results are in the extended version [1].

Lemma 4.1. *The function* $f : \mathrm{Runs}_X \to \mathrm{Runs}'$ *defined by*

$$f(x, s_0 \cdot a_0 \cdot s_1 \cdots) = (s_0, x) \cdot a_0 \cdot (s_1, x) \cdots$$

is an isomorphism between the measurable spaces $(\mathrm{Runs}_X, \mathrm{Cones}_X)$ *of the pMDP* M *and* $(\mathrm{Runs}', \mathrm{Cones}')$ *of its induced POMDP* M'. $\qquad\square$

This means that f is a bijection and both f and f^{-1} are measurable functions. In the sequel, we also write f for the bijection from Hist_X to Hist' defined in the same way. Note that f maps generators of Cones_X to generators of Cones' as

$$f(\{x\} \times \mathrm{Cone}(h_x)) = \mathrm{Cone}(f(x, h_x))$$

for any $x \in X$ and $h_x \in \mathrm{Hist}(M(x))$. Moreover, to state the obvious, f is related to the observation map O as follows: $O(f(x, h)) = h$.

Lemma 4.2. *There is a bijective correspondence* Φ *between the policies of the pMDP* M *and the policies of its induced POMDP* M' *given by* $\Phi(\pi_X) = \pi_X \circ f^{-1}$. *Its inverse acts as* $\Phi^{-1}(\pi') = \pi' \circ f$. $\qquad\square$

We are now able to relate the induced measures in a pMDP and its encoding.

Lemma 4.3. *Given a pMDP* $M = (S, A, X, T, i)$, *parameter distribution* p, *and policy* π_X, *we have*

$$\mathbb{P}^M_{\pi_X, i, p} = \mathbb{P}^{M'}_{\pi', i'} \circ f$$

where M' *is the induced POMDP of* M, $\pi' = \Phi(\pi_X)$, *and* $i'(s, x) = i(s) \cdot p(x)$. $\square$

Now the correctness of the encoding follows easily from the next result.

Theorem 4.2. *Given a pMDP* $M = (S, A, X, T, i)$, *parameter distribution* $p \in \mathcal{D}X$, *policy* π_X, *and an objective function* r, *we have*

$$\mathbb{E}^M_{\pi_X, i, p}(r) = \mathbb{E}^{M'}_{\pi', i'}(r') \tag{5}$$

where M' *is the induced POMDP of* M, $\pi' = \Phi(\pi_X)$, $r' = r \circ f^{-1}$, *and* $i'(s, x) = i(s) \cdot p(x)$. *The opposite also holds: Given a policy* π' *of the induced POMDP* M' *of* M, *Eq. (5) holds for* $\pi_X = \Phi^{-1}(\pi')$. $\qquad\square$

As a consequence, $\mathrm{Val}^M(p, r) = \mathrm{Val}^{M'}(r')$ and the policy π_X is expectation ε-optimal for M if and only if $\pi' = \Phi(\pi_X)$ is an ε-solution of M', for $\varepsilon \geq 0$.

5   Experiments[4]

In this section, we present several case studies to illustrate our approach of finding an expectation ε-optimal policy of a pMDP using an existing POMDP solver. The solver we use is AI-Toolbox [7], a well-known suite of algorithms for Markov models, which includes several algorithms for POMDPs with finite-horizon accumulated-reward objectives. For our case studies we evaluate two algorithms: *point-based value iteration* (PBVI) [34] and *Incremental Pruning* (IP) [10], for reasons explained in the extended version [1].

The PBVI implementation in AI-Toolbox does not generate beliefs on the fly (as in the anytime algorithm described in [34]), but generates a fixed number of beliefs upfront. First the simplex corners (i.e. the Diracs) and the midpoint are generated, then more are sampled (the point-based algorithms differ mainly in how the beliefs are sampled). Without at least the simplex corners and midpoint, the results are quite off - we chose to always pick at least all of those.

Experimental Setup. All the models that we analyze are described in PRISM file format. We use Storm [20] to parse the files and build the parametric model, that we then translate and pass to AI-Toolbox. Both Storm and AI-Toolbox have Python bindings allowing us to perform our encoding in Python. The experiments reported here ran on a NUMA machine with four 16-core 2.3 GHz AMD Opteron 6376 processors, 504 GB of main memory, and Linux kernel version 4.13.0. We used BenchExec [9] to run experiment series.

Selected Case Studies. We start by discussing a motivating *learner model* example that shows all important aspects. Then we present some results on a model of a robot moving on a grid and a consensus protocol model.

It is important to note here that many of the pMDP models studied (e.g., in the PARAM benchmarks [23]) exhibit only a weak form of nondeterminism, where the optimal policy does not depend on the parameters, i.e., the optimal policy is the same for any values of the parameters. Examples of such are, e.g., the Bounded Retransmission Protocol (BRP) and Zeroconf. Consensus is the exception, and we have solved some instances of Consensus as reported below. In the examples used in [35], showing how to obtain policies that optimize learning the parameter values, the optimal policy is again independent of the parameters.

Motivating Example – The (Repeated) Learner Model. Figure 1a shows the *learner model*, mentioned in [2], with initial state s and target state t, ignoring for the moment the grey (loop) transition labelled with action c.

After two steps we end up in state c, having visited state a with probability p and state b with probability $1 - p$. Here is the only choice in the model: between the actions a and b. Even though we assume p to be inaccessible, we can do better than flipping a coin: If we are given a prior belief over the parameter (an initial parameter distribution) we can use Bayesian inference to update this belief

---

[4] All of our code and models, as well as detailed results of the experiments can be found at http://github.com/sarming/pMDP-Toolbox.
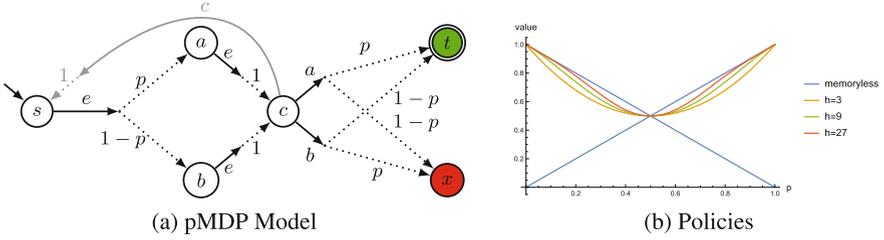
(a) pMDP Model   (b) Policies

Fig. 1. (Repeated) learner

with the information that either state a or b was visited. As a concrete example, assume we start with the uniform distribution over the parameter values as prior and state a was visited. Doing the calculation gives a posterior distribution with higher probability that the parameter value is close to 1 than to 0, suggesting that a is the better action to choose.

Let π be the policy that chooses action a in state c when state a was visited and action b when state b was visited. Figure 1b shows π (labelled $h = 3$) and the two memoryless policies (always a and always b). The policy π is clearly better in expectation than the two memoryless ones.

Actually, π is the optimal policy (among the 4 possible deterministic policies) when assuming a uniform parameter distribution. For a concrete parameter value $x \in [0, 1]$, the probability to reach t under π in the evaluated MDP $M(x)$ is $x^2 + (1 - x)^2$. Putting things together, using Eq. (4), we get that for a uniform parameter distribution the expected return is (a discrete approximation of) $\int_0^1 x^2 + (1 - x)^2 \, dx = 2/3$.

We consider the uniform distributions over 2, 3, 5, 10, 20, 50, 100, 200, 500, and 1000 evenly spaced points between 0 and 1. The expected return depends on the distribution: for just 2 points (the distribution assigning $1/2$ to 0 and $1/2$ to 1) the optimal policy provides probability 1 for reaching t; while for 1000 points the expected return is 0.6670. IP can solve all mentioned instances and generates π verifying that it is optimal.

The learner is inherently a finite-horizon model, as nothing happens after three steps. When we add the grey transition, we obtain the *repeated learner model* in which we can repeat the "experiment" getting closer and closer to the "sea surface" [2] given by the memoryless policies. The larger the horizon, the more experiments the ε-optimal policy runs. Only an odd number of experiments gives an actual improvement because we need a majority: having two experiments is as good as just having one. Therefore the value increases at $h = 3(2n + 1)$, see Fig. 1b and Table 1b.

Figure 2 shows the results of the experimental evaluation of the repeated learner with increasing horizon and up to 100 parameter points. IP provides better policies than PBVI with respect to the same number of points in the parameter space. However, it does not scale as well as PBVI and we had to drop it for more complex models.
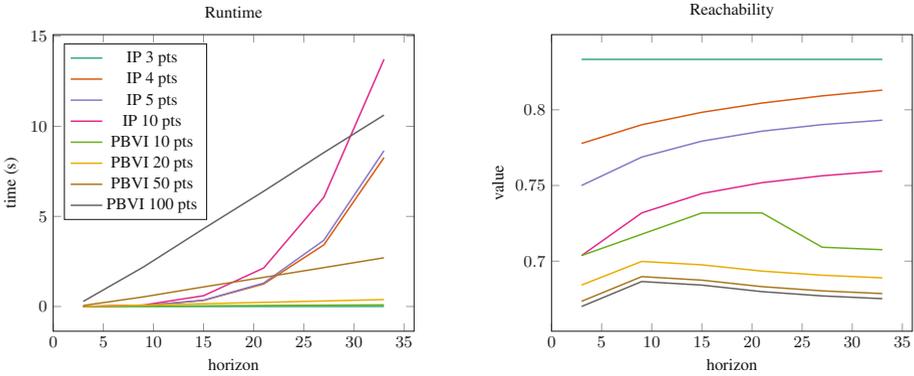
Table 1. IP results for (repeated) learner

points	states	time (s)	value	nodes
2	14	0.001	1	3
5	35	0.003	0.75	7
10	70	0.005	0.703	7
20	140	0.008	0.684	7
50	350	0.025	0.673	7
100	700	0.098	0.670	7
200	1400	0.423	0.668	7
500	3500	2.496	0.667	7
1000	7000	9.725	0.667	7

(a) $h = 3$ (i.e. without repeating)

h	time (s)	value	nodes
3	0.003	0.704	7
9	0.081	0.732	23
15	0.598	0.745	47
21	2.137	0.752	79
27	6.059	0.756	119
33	13.715	0.760	167
39	30.324	0.762	223

(b) 10 points

Fig. 2. IP and PBVI results for repeated learner

Robot on a Grid. Our next case study is a variant of the ever popular Grid-world [37]. The instance that we report on is a 3×3 grid with initial state at position $(1, 1)$, sink at position $(2, 2)$, and target at position $(3, 3)$. We want to maximize the probability to reach the target from the initial position.

The robot has (up to) 4 actions available: up, down, left, right. The actions are probabilistic in the sense that with some probability, instead of going forward the robot may end up in the cell to the left or to the right. We compare two variants: In both variants there is a parameter p that describes the total error probability. In the 1-parameter variant, the probability to err left and right is equal to $p/2$; In the 2-parameter variant we also include a left-right bias b, resulting in probability $p * b$ to err left and $p * (1 - b)$ to err right. If it is not possible to go left or right, then the other option gets probability p. For example, the action up in cell $(1, 1)$ leads correctly to cell $(2, 1)$ with probability $1 - p$ and to cell $(1, 2)$ on the right with probability p (as no cell is on the left), in both models. The action up in cell $(1, 2)$ has the possibility to err left and right, hence

shows the difference between the two models. See [2] for a detailed description of robot-on-a-grid models. In that terminology, we use the "fixed failure" variant.

Consensus. The consensus protocol model is the only PARAM benchmark [23] that has true nondeterminism in the sense that its policy depends on the parameter values. The protocol was introduced by Aspnes and Herlihy [3]. The 2-parameter model is exactly the same as the PARAM model, see [23] for all details. The 1-parameter model depends on a parameter p and is obtained by setting $p_1 = p$ and $p_2 = 1 - p$, i.e., it is a bias parameter with average $1/2$. We used $N = K = 2$ and the target state is the state in which consensus is reached with the preferred value.

Figures 3 and 4 show the experimental results of the robot on a grid, and the consensus protocol benchmarks, respectively. We only ran PBVI as IP was too slow and in both cases notice that the runtime grows exponentially with the horizon. The grid pMDP model has 10 states and 2 parameters, hence the
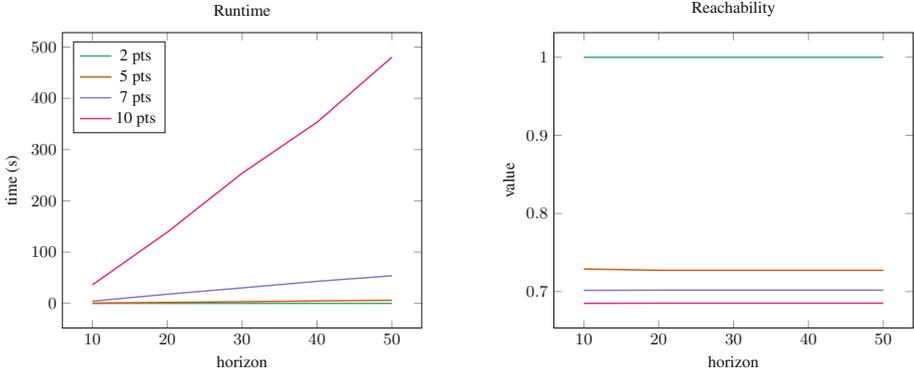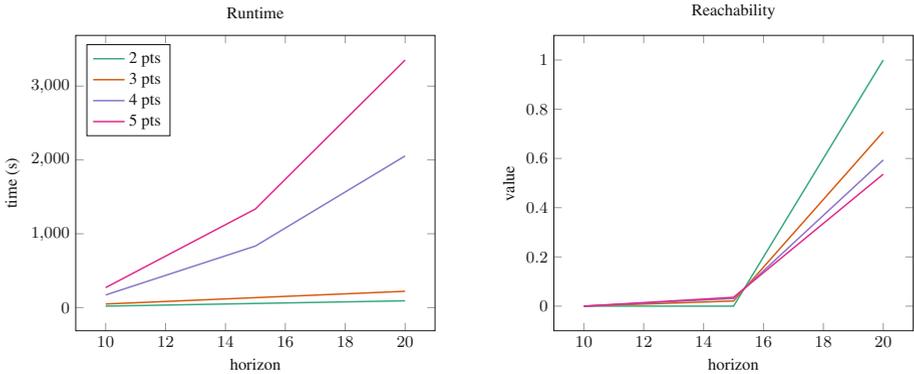

Fig. 3. PBVI results for robot on a grid

Fig. 4. PBVI results for consensus

induced POMDP for 10 parameter points has 1000 states. The consensus protocol model has 273 states and hence the induced POMDP for 5 parameter points has 1365 states.

6   Conclusion

We have presented a way to compute parameter-independent strategies that are expectation ε-optimal for pMDP by encoding the problem as to compute ε-solutions in POMDPs. We have implemented this approach using Storm [20] and AI-Toolbox [7] and we have evaluated on different case studies. Future work will focus on improving the efficiency of the current algorithms (for better scalability) by taking into account the particular POMDP structure resulting from the encoding.

References

1. Arming, S., Bartocci, E., Chatterjee, K., Katoen, J., Sokolova, A.: Parameter-independent strategies for pMDPs via POMDPs. arXiv 1806.05126 (2018). http://arxiv.org/abs/1806.05126
2. Arming, S., Bartocci, E., Sokolova, A.: SEA-PARAM: exploring schedulers in parametric MDPs. In: Proceedings of the QAPL 2017. EPTCS, vol. 250, pp. 25–38 (2017)
3. Aspnes, J., Herlihy, M.: Fast randomized consensus using shared memory. J. Algorithms **11**(3), 441–461 (1990)
4. Baier, C., Größer, M., Bertrand, N.: Probabilistic ω-automata. J. ACM **59**(1), 1:1–1:52 (2012)
5. Baier, C., Katoen, J.: Principles of Model Checking. MIT Press, Cambridge (2008)
6. Baldi, M., et al.: A probabilistic small model theorem to assess confidentiality of dispersed cloud storage. In: Bertrand, N., Bortolussi, L. (eds.) QEST 2017. LNCS, vol. 10503, pp. 123–139. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66335-7_8
7. Bargiacchi, E.: AI-Toolbox. https://github.com/Svalorzen/AI-Toolbox/
8. Bartocci, E., Grosu, R., Katsaros, P., Ramakrishnan, C.R., Smolka, S.A.: Model repair for probabilistic systems. In: Abdulla, P.A., Leino, K.R.M. (eds.) TACAS 2011. LNCS, vol. 6605, pp. 326–340. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19835-9_30
9. Beyer, D., Löwe, S., Wendler, P.: Benchmarking and resource measurement. In: Fischer, B., Geldenhuys, J. (eds.) SPIN 2015. LNCS, vol. 9232, pp. 160–178. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-23404-5_12
10. Cassandra, A.R., Littman, M.L., Zhang, N.L.: Incremental pruning - a simple, fast, exact method for partially observable Markov decision processes. In: Proceedings of the UAI 1997, pp. 54–61 (1997)
11. Chatterjee, K., Doyen, L., Henzinger, T.A.: Qualitative analysis of partially-observable Markov decision processes. In: Hliněný, P., Kučera, A. (eds.) MFCS 2010. LNCS, vol. 6281, pp. 258–269. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15155-2_24
12. Chatterjee, K., Chmelik, M.: POMDPs under probabilistic semantics. Artif. Intell. **221**, 46–72 (2015)

13. Chatterjee, K., Chmelik, M., Davies, J.: A symbolic SAT-based algorithm for almost-sure reachability with small strategies in POMDPs. In: Proceedings of the AAAI 2016, pp. 3225–3232 (2016)
14. Chatterjee, K., Chmelik, M., Gupta, R., Kanodia, A.: Optimal cost almost-sure reachability in POMDPs. Artif. Intell. **234**, 26–48 (2016)
15. Chatterjee, K., Doyen, L., Gimbert, H., Henzinger, T.A.: Randomness for free. In: Hliněný, P., Kučera, A. (eds.) MFCS 2010. LNCS, vol. 6281, pp. 246–257. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15155-2_23
16. Chen, T., Hahn, E.M., Han, T., Kwiatkowska, M.Z., Qu, H., Zhang, L.: Model repair for Markov decision processes. In: Proceedings of the TASE 2013, pp. 85–92 (2013)
17. Cubuktepe, M.: Sequential convex programming for the efficient verification of parametric MDPs. In: Legay, A., Margaria, T. (eds.) TACAS 2017. LNCS, vol. 10206, pp. 133–150. Springer, Heidelberg (2017). https://doi.org/10.1007/978-3-662-54580-5_8
18. Daws, C.: Symbolic and parametric model checking of discrete-time Markov chains. In: Liu, Z., Araki, K. (eds.) ICTAC 2004. LNCS, vol. 3407, pp. 280–294. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-31862-0_21
19. Dehnert, C., et al.: PROPhESY: a probabilistic parameter synthesis tool. In: Kroening, D., Păsăreanu, C.S. (eds.) CAV 2015. LNCS, vol. 9206, pp. 214–231. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-21690-4_13
20. Dehnert, C., Junges, S., Katoen, J.-P., Volk, M.: A STORM is coming: a modern probabilistic model checker. In: Majumdar, R., Kunčak, V. (eds.) CAV 2017. LNCS, vol. 10427, pp. 592–600. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63390-9_31
21. Hahn, E.M., Han, T., Zhang, L.: Probabilistic reachability for parametric Markov models. STTT **13**(1), 3–19 (2011)
22. Hahn, E.M., Han, T., Zhang, L.: Synthesis for PCTL in parametric Markov decision processes. In: Bobaru, M., Havelund, K., Holzmann, G.J., Joshi, R. (eds.) NFM 2011. LNCS, vol. 6617, pp. 146–161. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20398-5_12
23. Hahn, E.M., Hermanns, H., Zhang, L., Wachter, B.: PARAM case studies (2015). https://depend.cs.uni-saarland.de/tools/param/casestudies
24. Halmos, P.R.: Measure Theory. Springer, New York (1974). https://doi.org/10.1007/978-1-4684-9440-2
25. Jansen, N., et al.: Accelerating parametric probabilistic verification. In: Norman, G., Sanders, W. (eds.) QEST 2014. LNCS, vol. 8657, pp. 404–420. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10696-0_31
26. Junges, S., Jansen, N., Wimmer, R., Quatmann, T., Winterer, L., Katoen, J., Becker, B.: Finite-state controllers of POMDPs via parameter synthesis. In: Proceedings of the UAI 2018 (2018)
27. Kreinovich, V., Lakeyev, A., Rohn, J., Kahl, P.: Computational Complexity and Feasibility of Data Processing and Interval Computations, Applied Optimization, vol. 10. Springer, Boston (1998). https://doi.org/10.1007/978-1-4757-2793-7
28. Kwiatkowska, M., Norman, G., Parker, D.: PRISM 4.0: verification of probabilistic real-time systems. In: Gopalakrishnan, G., Qadeer, S. (eds.) CAV 2011. LNCS, vol. 6806, pp. 585–591. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22110-1_47
29. Lanotte, R., Maggiolo-Schettini, A., Troina, A.: Parametric probabilistic transition systems for system design and analysis. Form. Asp. Comput. **19**(1), 93–109 (2007)

30. Lukina, A., et al.: ARES: adaptive receding-horizon synthesis of optimal plans. In: Legay, A., Margaria, T. (eds.) TACAS 2017. LNCS, vol. 10206, pp. 286–302. Springer, Heidelberg (2017). https://doi.org/10.1007/978-3-662-54580-5_17

31. Madani, O., Hanks, S., Condon, A.: On the undecidability of probabilistic planning and related stochastic optimization problems. Artif. Intell. **147**(1–2), 5–34 (2003)

32. Medina Ayala, A.I., Andersson, S.B., Belta, C.: Probabilistic control from time-bounded temporal logic specifications in dynamic environments. In: Proceedings of the ICRA 2012, pp. 4705–4710. IEEE (2012)

33. Pathak, S., Ábrahám, E., Jansen, N., Tacchella, A., Katoen, J.-P.: A greedy approach for the efficient repair of stochastic models. In: Havelund, K., Holzmann, G., Joshi, R. (eds.) NFM 2015. LNCS, vol. 9058, pp. 295–309. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-17524-9_21

34. Pineau, J., Gordon, G.J., Thrun, S.: Point-based value iteration - an anytime algorithm for POMDPs. In: Proceedings of the IJCAI 2003, pp. 1025–1032 (2003)

35. Polgreen, E., Wijesuriya, V.B., Haesaert, S., Abate, A.: Automated experiment design for data-efficient verification of parametric Markov decision processes. In: Bertrand, N., Bortolussi, L. (eds.) QEST 2017. LNCS, vol. 10503, pp. 259–274. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66335-7_16

36. Quatmann, T., Dehnert, C., Jansen, N., Junges, S., Katoen, J.-P.: Parameter synthesis for Markov models: faster than ever. In: Artho, C., Legay, A., Peled, D. (eds.) ATVA 2016. LNCS, vol. 9938, pp. 50–67. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46520-3_4

37. Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach. Prentice Hall, Upper Saddle River (2009)

38. Sennott, L.I.: Stochastic Dynamic Programming and the Control of Queueing Systems. Wiley, New York (1998)

39. Spaan, M.T.J., Vlassis, N.: Perseus: randomized point-based value iteration for POMDPs. J. Artif. Intell. Res. **24**, 195–220 (2011)