Parameter Synthesis for Markov Models: Faster Than Ever

Tim Quatmann[1], Christian Dehnert[1], Nils Jansen[2], Sebastian Junges[1(✉)], and Joost-Pieter Katoen[1]

[1] RWTH Aachen University, Aachen, Germany
sebastian.junges@rwth-aachen.de
[2] University of Texas at Austin, Austin, USA

Abstract. We propose a conceptually simple technique for verifying probabilistic models whose transition probabilities are parametric. The key is to replace parametric transitions by nondeterministic choices of extremal values. Analysing the resulting parameter-free model using off-the-shelf means yields (refinable) lower and upper bounds on probabilities of regions in the parameter space. The technique outperforms the existing analysis of parametric Markov chains by several orders of magnitude regarding both run-time and scalability. Its beauty is its applicability to various probabilistic models. It in particular provides the first sound and feasible method for performing parameter synthesis of Markov decision processes.

1 Introduction

The key procedure in probabilistic model checking is computing reachability probabilities: What is the probability to reach some target state? For models exhibiting nondeterminism, such as Markov decision processes (MDPs), the probability to reach a state is subject to resolving the nondeterminism, and one considers minimal and maximal reachability probabilities. Model checkers support these procedures, e. g., `PRISM` [1] and `iscasMc` [2]. Successful applications to models of hundreds of millions of states have been reported, and extensions to stochastic games exist [3].

This paper treats *parameter synthesis* in Markov models. Given a model whose transition probabilities are (polynomials over) variables, and a reachability specification—e.g., the likelihood to reach a bad state should be below 10^{-6}—the parameter synthesis problem aims at finding all parameter values for which the parametric model satisfies the specification. In practise, this amounts to partition the parameter space into *safe* and *unsafe* regions with a large (say, $>95\,\%$) coverage. For a system in which components are subject to random failures, parameter synthesis is thus able to obtain the maximal tolerable failure probability of the components while ensuring the system's specification.

Parametric probabilistic models have various applications as witnessed by several recent works. Model repair [4] exploits parametric Markov chains (MCs) to tune the parameters of the model. In quality-of-service analysis of software,

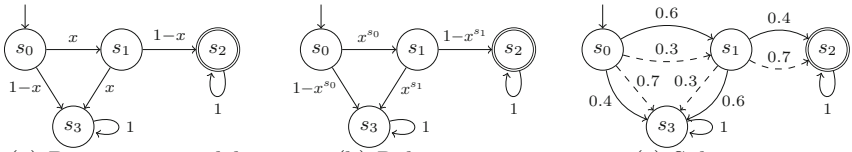
C. Artho et al. (Eds.): ATVA 2016, LNCS 9938, pp. 50–67, 2016.
DOI: 10.1007/978-3-319-46520-3_4

parameters are used to model the unquantified estimation errors in log data [5]. Ceska *et al.* [6] consider the problem of synthesising rate parameters in stochastic biochemical networks. Parametric probabilistic models are also used to rank patches in the repair of software [7] and for computing perturbation bounds [8, 9]. The main problem though is that current parametric probabilistic model-checking algorithms cannot cope with the complexity of these applications. Their scalability is restricted to a couple of thousands of states and a few (preferably independent) parameters, and models with nondeterminism are out of reach. (The only existing algorithm [10] for parametric MDPs uses an unsound heuristic in its implementation to improve scalability.)

We present an algorithm that overcomes all these limitations: It is scalable to millions of states, several (dependent) parameters, and—perhaps most importantly—provides the first sound and feasible technique to do parameter synthesis of parametric MDPs.

The key technique used so far is computing a rational function (in terms of the parameters) expressing the reachability probability in a parametric MC. Tools like `PARAM` [11], `PRISM` [1], and `PROPhESY` [12] exploit (variants of) the state elimination approach by Daws [13] to obtain such a function which conceptually allows for many types of analysis. While state elimination is feasible for millions of states [12], it does not scale well in the number of different parameters. Moreover, the size of the obtained functions often limits the practicability as analysing the (potentially large) rational function via SMT solving [12] is often not feasible.

This paper takes a completely different approach: *Parameter lifting*. Consider the parametric MC in Fig. 1(a) modelling two subsequent tosses of a biased coin, where the probability for *heads* is x. Inspired by an observation made in [14] on continuous time Markov chains, we first equip each state with a fresh parameter, thus removing parameter dependencies; the outcome (referred to as *relaxation*) is depicted in Fig. 1(b). Now, for each function over these state parameters, we compute *extremal values*, i. e., maximal and minimal probabilities. The key idea is to replace the (parametric) probabilistic choice at each state by a *nondeterministic choice* between these extremal values; we call this *substitution*. This is exemplified in Fig. 1(c), assuming *heads* has a likelihood in $[0.3, 0.6]$. The resulting (non-parametric) model can be verified using off-the-shelf, efficient algorithms. Applying this procedure to a parametric MC (as in the example) yields a parameter-free MDP. Parameter lifting thus boils down to verify an MDP and



(a) Parametric model (b) Relaxation (c) Substitution

Fig. 1. Two biased coin tosses and the specification "First *heads* then *tails*".

avoids computing rational functions and SMT solving. The beauty of this technique is that it can be applied to parametric MDPs without much further ado. Parameter lifting of a parametric MDP yields a parameter-free two-player stochastic game (SG). SGs and MDPs can be solved using techniques such as value and policy iteration. Note that the theoretical complexity for solving MDPs is lower than for SGs.

This paper presents the details of parameter lifting, and proves the correctness for parametric Markov models whose parameters are given as *multi-affine polynomials*. This covers a rich class of models, e. g., the diverse set of parametric benchmarks available at the `PARAM` webpage are of this form. Experiments demonstrate the feasibility: The parameter lifting approach can treat Markov models of millions of states with thousands of parametric transitions. This applies to parametric MCs as well as MDPs. Parameter lifting achieves a parameter space coverage of at least 95% rather quickly. This is out of reach for competitive techniques such as SMT-based [12] and sampling-based [10] parameter synthesis.

2 Preliminaries

Let V be a finite set of *parameters* over the domain $\mathbb{R}$ ranged over by x, y, z. A *valuation* for V is a function $u\colon V \to \mathbb{R}$. Let $\mathbb{Q}_V$ denote the set of *multi-affine multivariate polynomials* f over V satisfying $f = \sum_{i \leq m} a_i \cdot \prod_{x \in V_i} x$ for suitable $m \in \mathbb{N}$, $a_i \in \mathbb{Q}$, and $V_i \subseteq V$ (for $i \leq m$). $\mathbb{Q}_V$ does not contain polynomials where a variable has a degree greater than 1, e. g., $x \cdot y \in \mathbb{Q}_V$ but $x^2 \notin \mathbb{Q}_V$. We write $f = 0$ if f can be reduced to 0, and $f \neq 0$ otherwise. Applying the valuation u to $f \in \mathbb{Q}_V$ results in a real number $f[u] \in \mathbb{R}$, obtained from f by replacing each occurrence of variable x in f by $u(x)$.

2.1 Probabilistic Models

We consider different types of parametric (discrete) probabilistic models. They can all be seen as transition systems (with a possible partition of the state space into two sets) where the transitions are labeled with polynomials in $\mathbb{Q}_V$.

Definition 1 (Parametric Probabilistic Models). *A* parametric stochastic game (pSG) *is a tuple* $\mathfrak{M} = (S, V, s_I, Act, \mathcal{P})$ *with a finite set* S *of states such that* $S = S_\circ \uplus S_\square$*, a finite set* V *of parameters over* $\mathbb{R}$*, an initial state* $s_I \in S$*, a finite set* Act *of actions, and a transition function* $\mathcal{P}\colon S \times Act \times S \to \mathbb{Q}_V$ *satisfying:* $Act(s) \neq \emptyset$ *where* $Act(s) = \{\alpha \in Act \mid \exists s' \in S.\, \mathcal{P}(s, \alpha, s') \neq 0\}$.

- $\mathfrak{M}$ *is a* parametric Markov decision process (pMDP) *if* $S_\circ = \emptyset$ *or* $S_\square = \emptyset$.
- *pMDP* $\mathfrak{M}$ *is a* parametric Markov chain (pMC) *if* $|Act(s)| = 1$ *for all* $s \in S$.

We will refer to pMCs by $\mathcal{D}$, to pMDPs by $\mathcal{M}$ and to pSGs by $\mathcal{G}$. pSGs are two-player parametric stochastic games involving players $\circ$ and $\square$ with states

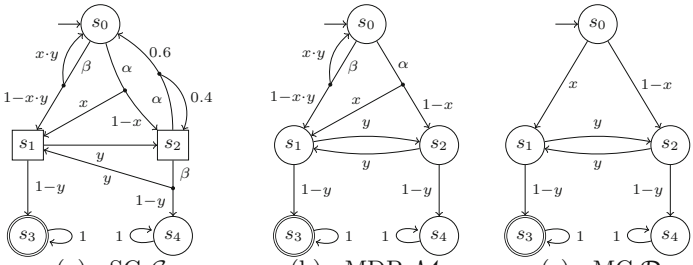


Fig. 2. The considered types of parametric probabilistic models.

in $S_\circ$ and $S_\square$, respectively, whose transition probabilities are represented by polynomials from $\mathbb{Q}_V$. The players *nondeterministically* choose an action at each state and the successors are intended to be determined *probabilistically* as defined by the transition function. $Act(s)$ is the set of *enabled* actions at state s. As $Act(s)$ is non-empty for all $s \in S$, there are no deadlock states. For state s and action α, we set $V_s^\alpha = \{x \in V \mid x \text{ occurs in } \mathcal{P}(s, \alpha, s') \text{ for some } s' \in S\}$.

pMDPs and pMCs are one- and zero-player parametric stochastic games, respectively. As pMCs have in fact just a single enabled action at each state, we omit this action in the notation and just write $\mathcal{P}(s, s')$ and V_s.

Example 1. Figure 2 depicts (a.) a pSG, (b.) a pMDP, and (c.) a pMC with parameters $\{x, y\}$. The states of the players $\circ$ and $\square$ are depicted with circles and rectangles, respectively. The initial state is indicated by an arrow; target states have double lines. We draw a transition from state s to s' and label it with α and $\mathcal{P}(s, \alpha, s')$ whenever $\mathcal{P}(s, \alpha, s') \neq 0$. If $|Act(s)| = 1$, the action is omitted.

Remark 1. In the literature [12,15], the images of transition functions (of pMCs) are rational functions, i. e., fractions of polynomials. This is mainly motivated by the usage of state elimination for computing functions expressing reachability probabilities. As our approach does not rely on state elimination, the set of considered functions can be simplified. The restriction to polynomials in $\mathbb{Q}_V$ is realistic; *all* benchmarks from the `PARAM` webpage [16] are of this form. We will exploit this restriction in our proof of Theorem 1.

Definition 2 (Stochastic Game). *A pSG $\mathcal{G}$ is a* stochastic game (SG) *if* $\mathcal{P} \colon S \times Act \times S \to [0, 1]$ *and* $\sum_{s' \in S} \mathcal{P}(s, \alpha, s') = 1$ *for all* $s \in S$ *and* $\alpha \in Act(s)$.

Analogously, MCs and MDPs are defined as special cases of pMCs and pMDPs. Thus, a model is *parameter-free* if all transition probabilities are constant.

Valuations and Rewards. Applying a *valuation* u to parametric model $\mathfrak{M}$, denoted $\mathfrak{M}[u]$, replaces each polynomial f in $\mathfrak{M}$ by $f[u]$. We call $\mathfrak{M}[u]$ the

instantiation of $\mathfrak{M}$ at u. The typical application of u is to replace the transition function f by the probability $f[u]$. A valuation u is *well-defined* for $\mathfrak{M}$ if the replacement yields probability distributions, i. e., if $\mathfrak{M}[u]$ is an MC, an MDP, or an SG, respectively.

Parametric probabilistic models are extended with *rewards* (or dually, costs) by adding a *reward function* $\mathrm{rew}\colon S \to \mathbb{Q}_V$ which assigns rewards to states of the model. Intuitively, the reward $\mathrm{rew}(s)$ is earned upon leaving the state s.

Schedulers. The nondeterministic choices of actions in pSGs and pMDPs can be resolved using *schedulers*[1]. In our setting it suffices to consider memoryless deterministic schedulers [17]. For more general definitions we refer to [18].

Definition 3 (Scheduler). *A* scheduler *for pMDP* $\mathcal{M} = (S, V, s_I, Act, \mathcal{P})$ *is a function* $\sigma\colon S \to Act$ *with* $\sigma(s) \in Act(s)$ *for all* $s \in S$.

Let $\mathfrak{S}(\mathcal{M})$ denote the set of all schedulers for $\mathcal{M}$. Applying a scheduler to a pMDP yields an *induced parametric Markov chain*, as all nondeterminism is resolved, i. e., the transition probabilities are obtained w. r. t. the choice of actions.

Definition 4 (Induced pMC). *For pMDP* $\mathcal{M} = (S, V, s_I, Act, \mathcal{P})$ *and scheduler* $\sigma \in \mathfrak{S}(\mathcal{M})$*, the* pMC induced by $\mathcal{M}$ and σ *is* $\mathcal{M}^\sigma = (S, V, s_I, \mathcal{P}^\sigma)$ *where*

$$\mathcal{P}^\sigma(s, s') = \mathcal{P}(s, \sigma(s), s') \quad \text{for all } s, s' \in S \ .$$

Resolving nondeterminism in an SG requires to have individual schedulers for each player. For $S_\circ$ and $S_\square$ we need schedulers $\sigma \in \mathfrak{S}_\circ(\mathcal{G})$ and $\rho \in \mathfrak{S}_\square(\mathcal{G})$ of the form $\sigma\colon S_\circ \to Act$ and $\rho\colon S_\square \to Act$. The induced pMC $\mathcal{G}^{\sigma,\rho}$ of a pSG $\mathcal{G}$ with schedulers σ and ρ for both players is defined analogously to the one for pMDPs.

Example 2. Reconsider the models $\mathcal{G}$, $\mathcal{M}$, and $\mathcal{D}$ as shown in Fig. 2. For schedulers σ, ρ with $\sigma(s_0) = \alpha$ and $\rho(s_2) = \beta$, the induced pMCs satisfy $\mathcal{G}^{\sigma,\rho} = \mathcal{M}^\rho = \mathcal{D}$.

2.2 Properties of Interest

As specifications we consider *reachability properties* and *expected reward properties*. We first define these properties on MCs and then discuss the other models.

Properties on MCs. For MC $\mathcal{D}$ with state space S, let $\Pr_s^{\mathcal{D}}(\Diamond T)$ denote the probability to reach a set of target states $T \subseteq S$ from state $s \in S$ within $\mathcal{D}$; simply $\Pr^{\mathcal{D}}(\Diamond T)$ refers to this specific probability for the initial state s_I. We use a standard probability measure on infinite paths through an MC as defined in [18, Ch. 10]. For threshold $\lambda \in [0, 1]$, the *reachability property* asserting that a target state is to be reached with probability at most λ is denoted $\varphi_{reach} = \mathbb{P}_{\leq\lambda}(\Diamond T)$. The property is satisfied by $\mathcal{D}$, written $\mathcal{D} \models \varphi_{reach}$, iff $\Pr^{\mathcal{D}}(\Diamond T) \leq \lambda$. (Comparisons like $<$, $>$, and $\geq$ are treated in a similar way.)

[1] Also referred to as adversaries, strategies, or policies.

The reward of a path through an MC $\mathcal{D}$ until T is the sum of the rewards of the states visited along on the path before reaching T. The expected reward of a finite path is given by its probability times its reward. Given $\Pr^{\mathcal{D}}(\Diamond T) = 1$, the expected reward of reaching $T \subseteq S$, is the sum of the expected rewards of all paths to reach T. An expected reward property is satisfied if the expected reward of reaching T is bounded by a threshold $\kappa \in \mathbb{R}$. Formal definitions can be found in e.g., [18, ch. 10].

Properties on Nondeterministic Models. In order to define a probability measure for MDPs and SGs, the nondeterminism has to be resolved. A reachability property $\mathbb{P}_{\leq\lambda}(\Diamond T)$ is satisfied for an MDP $\mathcal{M}$ iff it holds for all induced MCs:

$$\mathcal{M} \models \mathbb{P}_{\leq\lambda}(\Diamond T) \iff \big(\max_{\sigma \in \mathfrak{S}(\mathcal{M})} \Pr^{\mathcal{M}^\sigma}(\Diamond T)\big) \leq \lambda.$$

Satisfaction of a property φ for an SG $\mathcal{G}$ depends on the objectives of both players. We write $\mathcal{G} \models_{\triangle} \varphi$ iff players in $\triangle \subseteq \{\circ, \square\}$ can enforce that φ holds, e. g.,

$$\mathcal{G} \models_{\{\circ\}} \mathbb{P}_{\leq\lambda}(\Diamond T) \iff \big(\min_{\sigma \in \mathfrak{S}_\circ(\mathcal{G})} \max_{\rho \in \mathfrak{S}_\square(\mathcal{G})} \Pr^{\mathcal{G}^{\sigma,\rho}}(\Diamond T)\big) \leq \lambda.$$

Computing the maximal (or minimal) probability to reach a set of target states from the initial state can be done using standard techniques, such as linear programming, value iteration or policy iteration [19].

The satisfaction relation for expected reward properties is defined analogously. As usual, we write $\mathfrak{M} \models \neg\varphi$ whenever $\mathfrak{M} \not\models \varphi$.

3 Regional Model Checking of Markov Chains

In the following, we consider sets of valuations that map each parameter to a value within a given interval. We present an approximative approach to check all instantiations of a pMC with respect to a valuation in such a set. This consists of three steps: Formalising regions and the considered problem, construction of the sound over-approximation, and reduction to an MDP problem.

3.1 Regions

Definition 5 (Region). *Given a set of parameters $V = \{x_1, \dots x_n\}$ and rational* parameter bounds $B(x_i) = \{b_1, b_2\}$. *The parameter bounds induce a* parameter interval $I(x_i) = [b_1, b_2]$ *with* $b_1 \leq b_2$. *The set of valuations* $\{u \mid \forall x_i \in V.\, u(x_i) \in I(x_i)\}$ *is called a* region (for V).

The regions we consider correspond to $_{x \in V} I(x)$, i. e., they are *hyperrectangles*.

We aim to identify sets of instantiated models by regions. That is, regions represent instantiations $\mathfrak{M}[u]$ of a parametric model $\mathfrak{M}$. As these instantiations are only well-defined under some restrictions, we lift these restrictions to regions.

Definition 6 (Well-Defined Region). *Let $\mathfrak{M}$ be a parametric model. A region r for V is* well-defined *for $\mathfrak{M}$ if for all $u \in r$ it holds that u is well-defined for $\mathfrak{M}$, and for all polynomials f in $\mathfrak{M}$ either $f = 0$ or $f[u] > 0$.*

The first condition says that $\mathfrak{M}[u]$ is a probabilistic model (SG, MC, or MDP) while the second one ensures that $\mathfrak{M}[u]$ and $\mathfrak{M}$ have the same topology.

Example 3. Let $\mathcal{D}$ be the pMC in Fig. 3(a), the region $r = [0.1, 0.8] \times [0.4, 0.7]$ and the valuation $u = (0.8, 0.6) \in r$. Figure 3(b) depicts the instantiation $\mathcal{D}[u]$, which is an MC as defined in Sect. 2.1 with the same topology as $\mathcal{D}$. As this holds for all possible instantiations $\mathcal{D}[u']$ with $u' \in r$, region r is well-defined. The region $r' = [0, 1] \times [0, 1]$ is not well-defined as, e. g., the valuation $(0, 0) \in r'$ results in an MC that has no transition from s_1 to s_2.

Our aim is to prove that a property φ holds *for all instantiations* of a parametric model $\mathfrak{M}$ which are represented by a region r, i. e., $\mathfrak{M}, r \models \varphi$ defined as follows.

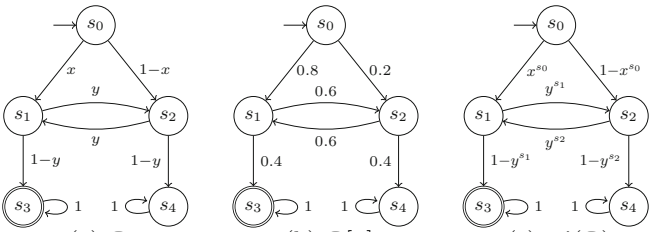


Fig. 3. A pMC $\mathcal{D}$, some instantiation $\mathcal{D}[u]$ and the relaxation $\mathsf{rel}(\mathcal{D})$.

Definition 7 (Satisfaction Relation for Regions). *For a parametric model $\mathfrak{M}$, a well-defined region r, and a property φ, the relation $\models$ is defined as*

$$\mathfrak{M}, r \models \varphi \iff \mathfrak{M}[u] \models \varphi \text{ for all } u \in r.$$

Notice that $\mathfrak{M}, r \not\models \varphi$ implies $\mathfrak{M}[u] \not\models \varphi$ for *some* $u \in r$. This differs from $\mathfrak{M}, r \models \neg\varphi$ which implies $\mathfrak{M}[u] \not\models \varphi$ for *all* $u \in r$. If $\mathfrak{M}$ and φ are clear from the context, we will call region r *safe* if $\mathfrak{M}, r \models \varphi$ and *unsafe* if $\mathfrak{M}, r \models \neg\varphi$.

Let $\mathcal{D} = (S, V, s_I, \mathcal{P})$ be a pMC, r a region that is *well-defined for* $\mathcal{D}$, and $\varphi_{reach} = \mathbb{P}_{\leq\lambda}(\Diamond T)$ a reachability property. We want to infer that r is safe (or unsafe). We do this by considering the *maximal (or minimal)* possible reachability probability over all valuations u from r. We give the equivalences for safe regions:

$$\mathcal{D}, r \models \varphi_{reach} \iff \left(\max_{u \in r} \Pr^{\mathcal{D}[u]}(\Diamond T)\right) \leq \lambda$$

$$\mathcal{D}, r \models \neg\varphi_{reach} \iff \left(\min_{u \in r} \Pr^{\mathcal{D}[u]}(\Diamond T)\right) > \lambda$$

Remark 2. As shown in [13], $\Pr^{\mathcal{D}[u]}(\Diamond T)$ can be expressed as a rational function $f = g_1/g_2$ with polynomials g_1, g_2. As r is well-defined, $g_2(u) \neq 0$ for all $u \in r$. Therefore, f is continuous on the closed set r. Hence, there is always a valuation that induces the maximal (or minimal) reachability probability:

$$\sup_{u \in r} \Pr^{\mathcal{D}[u]}(\Diamond T) = \max_{u \in r} \Pr^{\mathcal{D}[u]}(\Diamond T)$$
$$\text{and} \quad \inf_{u \in r} \Pr^{\mathcal{D}[u]}(\Diamond T) = \min_{u \in r} \Pr^{\mathcal{D}[u]}(\Diamond T) \ .$$

Example 4. Reconsider the pMC $\mathcal{D}$ in Fig. 3(a) and region $r = [0.1, 0.8] \times [0.4, 0.7]$. We look for a valuation $u \in r$ that maximises $\Pr^{\mathcal{D}[u]}(\Diamond\{s_3\})$, i. e., the probability to reach s_3 from s_0. Notice that s_4 is the only state from which we cannot reach s_3, furthermore, s_4 is only reachable via s_2. Hence, it is best to avoid s_2. For the parameter x it follows that the value $u(x)$ should be as high as possible, i. e., $u(x) = 0.8$. Consider state s_1: As we want to reach s_3, the value of y should be preferably low. On the other hand, from s_2, y should be assigned a high value as we want to avoid s_4. Thus, it requires a thorough analysis to find an optimal value for y, due to the trade-off for the reachability probabilities from s_1 and s_2.

3.2 Relaxation

The idea of our approach, inspired by [14], is to drop these dependencies by means of a *relaxation* of the problem in order to ease finding an optimal valuation.

Definition 8 (Relaxation). *The* relaxation *of pMC* $\mathcal{D} = (S, V, s_I, \mathcal{P})$ *is the pMC* $\mathsf{rel}(\mathcal{D}) = (S, \mathsf{rel}_{\mathcal{D}}(V), s_I, \mathcal{P}')$ *with* $\mathsf{rel}_{\mathcal{D}}(V) = \{x_i^s \mid x_i \in V, s \in S\}$ *and* $\mathcal{P}'(s, s') = \mathcal{P}(s, s')[x_1, \ldots, x_n / x_1^s, \ldots, x_n^s]$.

Intuitively, the relaxation $\mathsf{rel}(\mathcal{D})$ arises from $\mathcal{D}$ by equipping each state with its own parameters and thereby eliminating parameter dependencies. We extend a valuation u for $\mathcal{D}$ to the *relaxed valuation* $\mathsf{rel}_{\mathcal{D}}(u)$ for $\mathsf{rel}(\mathcal{D})$ by $\mathsf{rel}_{\mathcal{D}}(u)(x_i^s) = u(x_i)$ for every s. We have that for all u, $\mathcal{D}[u] = \mathcal{D}[\mathsf{rel}_{\mathcal{D}}(u)]$. We lift the relaxation to regions such that $B(x_i^s) = B(x_i)$ for all s, i. e., $\mathsf{rel}_{\mathcal{D}}(r) = {}_{x_i^s \in \mathsf{rel}_{\mathcal{D}}(V)}\, B(x_i)$. We drop the subscript $\mathcal{D}$, whenever it is clear from the context.

Example 5. Fig. 3(c) depicts the relaxation $\mathsf{rel}(\mathcal{D})$ of the pMC $\mathcal{D}$ from Fig. 3(a). For $r = [0.1, 0.8] \times [0.4, 0.7]$ and $u = (0.8, 0.6) \in r$ from Example 3, we obtain $\mathsf{rel}(r) = [0.1, 0.8] \times [0.4, 0.7] \times [0.4, 0.7]$ and $\mathsf{rel}(u) = (0.8, 0.6, 0.6)$. The instantiation $\mathsf{rel}(\mathcal{D})[\mathsf{rel}(u)]$ corresponds to $\mathcal{D}[u]$ as depicted in Fig. 3(b). Notice that the relaxed region $\mathsf{rel}(r)$ contains also valuations, e.g., $(0.8, 0.5, 0.6)$ which give rise to instantiations which are not realisable by valuations in r.

For a pMC $\mathcal{D}$ and a region r that is well-defined for $\mathcal{D}$, notice that $\{\mathcal{D}[u] \mid u \in r\} \subseteq \{\mathsf{rel}(\mathcal{D})[u] \mid u \in \mathsf{rel}(r)\}$. Due to the fact that $\mathsf{rel}(\mathcal{D})$ is an over-approximation of $\mathcal{D}$, the maximal reachability probability over all instantiations of $\mathcal{D}$ within r is at most as high as the one for all instantiations of $\mathsf{rel}(\mathcal{D})$ within $\mathsf{rel}(r)$.

Lemma 1. *For pMC $\mathcal{D}$ and well-defined region r, we have*

$$\max_{u \in r} \left(\Pr^{\mathcal{D}[u]}(\Diamond T)\right) \;=\; \max_{u \in r} \left(\Pr^{\mathsf{rel}(\mathcal{D})[\mathsf{rel}(u)]}(\Diamond T)\right) \;\leq\; \max_{u \in \mathsf{rel}(r)} \left(\Pr^{\mathsf{rel}(\mathcal{D})[u]}(\Diamond T)\right).$$

Thus, if the relaxation satisfies a reachability property, so does the original pMC.

Corollary 1. *Given a pMC $\mathcal{D}$ and a well-defined region r it holds that*

$$\max_{u \in \mathsf{rel}(r)} \left(\Pr^{\mathsf{rel}(\mathcal{D})[u]}(\Diamond T)\right) \leq \lambda \text{ implies } \mathcal{D}, r \models \mathbb{P}_{\leq \lambda}(\Diamond T).$$

Note that the relaxation does not aggravate the problem for our setting. In fact, although $\mathsf{rel}(\mathcal{D})$ has (usually) much more parameters than $\mathcal{D}$, it is intuitively easier to find a valuation $u \in \mathsf{rel}(r)$ that maximises the reachability probability: For some $x_i^s \in \mathsf{rel}(V)$, we can always pick a value in $I(x_i^s)$ that maximises the probability to reach T from state s. There is no (negative) effect for the reachability probability at the remaining states as x_i^s only occurs at s.

Recall that the functions f occurring in $\mathsf{rel}(\mathcal{D})$ are of the form $f = \sum_{i \leq m} a_i \cdot \prod_{x \in V_i} x$ (with $a_i \in \mathbb{Q}$ and $V_i \subseteq \mathsf{rel}(V)$). Finding a valuation that maximises the reachability probability becomes especially easy for this setting: We only need to consider valuations u that set the value of each parameter to either the lowest or highest possible value, i. e., $u(x_i^s) \in B(x_i^s)$ for all $x_i^s \in \mathsf{rel}(V)$. This important result is stated as follows.

Theorem 1. *Let $\mathcal{D}$ be a pMC, r be a well-defined region, and $T \subseteq S$ be a set of target states. There is a valuation $u' \in \mathsf{rel}(r)$ satisfying $u'(x_i^s) \in B(x_i^s)$ for all $x_i^s \in \mathsf{rel}(V)$ such that $\Pr^{\mathsf{rel}(\mathcal{D})[u']}(\Diamond T) = \max_{u \in \mathsf{rel}(r)} \Pr^{\mathsf{rel}(\mathcal{D})[u]}(\Diamond T)$.*

We prove this by showing that any valuation which assigns some variable to something other than its bound can be modified such that the variable is assigned to its bound, without decreasing the induced reachability probability. The full proof including an illustrating example is given in [20].

3.3 Substituting Parameters with Nondeterminism

We have now seen that, in order to determine $\max_{u \in \mathsf{rel}(r)} \Pr^{\mathsf{rel}(\mathcal{D})[u]}(\Diamond T)$, we have to make a discrete choice over valuations $v \colon \mathsf{rel}(V) \to \mathbb{R}$ with $v(x_i^s) \in B(x_i)$. This choice can be made locally at every state, which brings us to the key idea of *constructing a (non-parametric) MDP out of the pMC $\mathcal{D}$ and the region r*, where nondeterministic choices represent all valuations that need to be considered.

Definition 9 (Substitution-pMC). *An MDP $\mathsf{sub}_r(\mathcal{D}) = (S, s_I, Act_{\mathsf{sub}}, \mathcal{P}_{\mathsf{sub}})$ is the* (parameter-)substitution of a pMC $\mathcal{D} = (S, V, s_I, \mathcal{P})$ and a region r *if* $Act_{\mathsf{sub}} = \biguplus_{s \in S} \{v \colon V_s \to \mathbb{R} \mid v(x_i) \in B(x_i)\}$ *and*

$$\mathcal{P}_{\mathsf{sub}}(s, v, s') = \begin{cases} \mathcal{P}(s, s')[v] & \text{if } v \in Act(s), \\ 0 & \text{otherwise.} \end{cases}$$

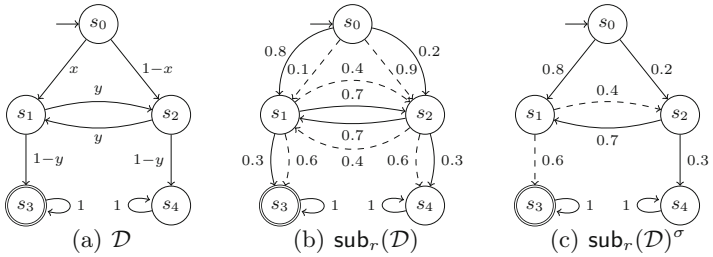


Fig. 4. Illustrating parameter-substitution.

Thus, choosing action v in s corresponds to assigning the extremal values $B(x_i)$ to the parameters x_i^s. The number of outgoing actions for s is therefore $2^{|V_s|}$.

Example 6. Consider pMC $\mathcal{D}$ – depicted in Fig. 4(a) – with $r = [0.1, 0.8] \times [0.4, 0.7]$ as before. The substitution of $\mathcal{D}$ on r is shown in Fig. 4(b). In $\mathcal{D}$, each outgoing transition of states s_0, s_1, s_2 is replaced by a nondeterministic choice in $\mathsf{sub}_r(\mathcal{D})$. That is, we either pick the upper or lower bound for the corresponding variable. The solid (dashed) lines depict transitions that belong to the action for the upper (lower) bound. For the states s_3 and s_4 there is no choice, as their outgoing transitions in $\mathcal{D}$ are constant. Figure 4(c) depicts the MC $\mathsf{sub}_r(\mathcal{D})^\sigma$ which is induced by the scheduler σ on $\mathsf{sub}_{\mathcal{D}}(r)$ that chooses the upper bounds at s_0 and s_2, and the lower bound at s_1. Notice that $\mathsf{sub}_r(\mathcal{D})^\sigma$ coincides with $\mathsf{rel}(\mathcal{D})[v]$ for a suitable valuation v, as depicted in Fig. 3(c).

First, observe that the nondeterministic choices introduced by the substitution only depend on the values $B(x_i)$ of the parameters x_i in r. Since the ranges of the parameters x_i^s in $\mathsf{rel}(r)$ agree with the range of x_i in r, we have

$$\mathsf{sub}_{\mathsf{rel}(r)}(\mathsf{rel}(\mathcal{D})) = \mathsf{sub}_r(\mathcal{D}) \quad \text{for all well-defined } r. \tag{1}$$

Second, note that the substitution encodes the local choices for a relaxed pMC. That is, for an arbitrary pMC, there is a one-to-one correspondence between schedulers $\sigma \in \mathfrak{S}(\mathsf{sub}_{\mathsf{rel}(r)}(\mathsf{rel}(\mathcal{D})))$ and valuations $v \in \mathsf{rel}(r)$ for $\mathsf{rel}(\mathcal{D})$ with $v(x_i^s) \in B(x_i)$. Combining the observations with Theorem 1, yields the following.

Corollary 2. *For a pMC $\mathcal{D}$, a well-defined region r and a set of target states T of $\mathcal{D}$:*

$$\max_{u \in r} \Pr^{\mathcal{D}[u]}(\Diamond T) \;\leq\; \max_{\sigma \in \mathfrak{S}} \Pr^{\mathsf{sub}_{\mathsf{rel}(r)}(\mathsf{rel}(\mathcal{D}))^\sigma}(\Diamond T) \;=\; \max_{\sigma \in \mathfrak{S}} \Pr^{\mathsf{sub}_r(\mathcal{D})^\sigma}(\Diamond T)$$
$$\min_{u \in r} \Pr^{\mathcal{D}[u]}(\Diamond T) \;\geq\; \min_{\sigma \in \mathfrak{S}} \Pr^{\mathsf{sub}_{\mathsf{rel}(r)}(\mathsf{rel}(\mathcal{D}))^\sigma}(\Diamond T) \;=\; \min_{\sigma \in \mathfrak{S}} \Pr^{\mathsf{sub}_r(\mathcal{D})^\sigma}(\Diamond T)$$

As a direct consequence of this, we can state Theorem 2.

Theorem 2. *Let $\mathcal{D}$ be a pMC, r be a well-defined region. Then*

$$\mathsf{sub}_r(\mathcal{D}) \models \mathbb{P}_{\leq\lambda}(\Diamond T) \text{ implies } \mathcal{D}, r \models \mathbb{P}_{\leq\lambda}(\Diamond T) \text{ and}$$
$$\mathsf{sub}_r(\mathcal{D}) \models \mathbb{P}_{>\lambda}(\Diamond T) \text{ implies } \mathcal{D}, r \models \neg\mathbb{P}_{\leq\lambda}(\Diamond T).$$

Hence, we can deduce whether $\mathcal{D}, r \models \varphi$ by applying standard techniques for MDP model checking to $\mathsf{sub}_r(\mathcal{D})$. If the over-approximation is too coarse for a conclusive answer, regions can be refined (cf. Sect. 5). Moreover, while the relaxation is key for showing the correctness, Eq. (1) proves that this step does not actually need to be performed.

Example 7. Reconsider Example 6. From $\mathsf{sub}_r(\mathcal{D})$ in Fig. 4(b), we can derive $\max_{\sigma\in\mathfrak{S}} \Pr^{\mathsf{sub}_r(\mathcal{D})^\sigma}(\Diamond T) = 47/60$ and, by Theorem 2, $\mathcal{D}, r \models \mathbb{P}_{\leq 0.8}(\Diamond T)$ follows. Despite the large considered region, we were able to establish a non-trivial upper bound on the reachability probability over all valuations in r.

Expected Reward Properties. The notions above can be applied to perform regional model checking of pMCs and expected reward properties. Regions have to be further restricted such that: $\Pr^{\mathcal{D}[u]}(\Diamond T) = 1$ for all $u \in r$ – to ensure that the expected reward is defined – and, for transition-rewards, reward-parameters and probability-parameters have to be disjoint. We can then generalise relaxation and substitution to the reward models, and obtain analogous results.

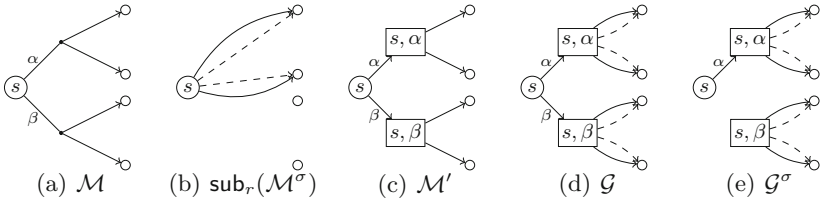
4 Regional Checking of Models with Nondeterminism

In the last section we showed how to bound reachability probabilities of pMCs from below and above. Introducing nondeterministic choices between these bounds enabled to utilise standard MDP model checking for the parameter synthesis. This approach can readily be generalised to systems originally exhibiting nondeterminism. In particular, for pMDPs this adds choices over valuations (inherent to parameters) to the choices over actions (inherent to MDPs). This new nondeterminism leads to a game with two players: One for the nondeterminism of the MDP and one for the abstracted parameters, yielding a stochastic game.

In the following, let $\mathcal{M} = (S, V, s_I, Act, \mathcal{P})$ be a pMDP and r a well-defined region for $\mathcal{M}$. We want to analyse r for all scheduler-induced pMCs $\mathcal{M}^\sigma$ of $\mathcal{M}$.

Example 8. Consider the pMDP $\mathcal{M}$ in Fig. 5(a), where state s has two enabled actions α and β. The scheduler σ given by $\{s \mapsto \alpha\}$ applied to $\mathcal{M}$ yields a pMC, which is subject to substitution, cf. Fig. 3(b).

The parameter substitution of a pMDP (cf. Fig. 5(a)) yields an SG—as in Fig. 5(d). It represents, for all schedulers of the pMDP, the substitution of each induced pMC. For the construction of the substitution, we first introduce intermediate states to separate nondeterministic actions from probabilistic choices in two steps:



(a) $\mathcal{M}$ (b) $\mathsf{sub}_r(\mathcal{M}^\sigma)$ (c) $\mathcal{M}'$ (d) $\mathcal{G}$ (e) $\mathcal{G}^\sigma$

Fig. 5. Illustration of the substitution of a pMDP.

- Split each state $s \in S$ into $\{s\} \uplus \{\langle s, \alpha\rangle \mid \alpha \in Act(s)\}$.
- For $s \in S$ and $\alpha \in Act(s)$, add a transition with probability one from s to $\langle s, \alpha\rangle$ and move the probabilistic choice at s w. r. t. α to $\langle s, \alpha\rangle$.

We obtain a pMDP as in Fig. 5(c) where state s has pure *nondeterministic* choices leading to states of the form $\langle s, \alpha\rangle$ with pure *probabilistic* choices. The subsequent substitution on the probabilistic states yields the stochastic game, where one player represents the nondeterminism of the original pMDP, while the other player decides whether parameters should be set to their lower or upper bound. Formally, the game $\mathcal{G} = \mathsf{sub}_r(\mathcal{M})$ is defined as follows.

Definition 10 (Substitution-pMDP). *Given a pMDP* $\mathcal{M} = (S, V, s_I, Act, \mathcal{P})$ *and a region* r*, an SG* $\mathsf{sub}_r(\mathcal{M}) = (S_\circ \uplus S_\square, s_I, Act_{\mathsf{sub}}, \mathcal{P}_{\mathsf{sub}})$ *with* $S_\circ = S$ *and* $S_\square = \{\langle s, \alpha\rangle \mid \alpha \in Act(s)\}$ *is the* (parameter-)substitution of $\mathcal{M}$ and r *if* $Act_{\mathsf{sub}} = Act \uplus \left(\biguplus_{\langle s,\alpha\rangle \in S_\square} Act_s^\alpha\right)$ *with* $Act_s^\alpha = \{v \colon V_s^\alpha \to \mathbb{R} \mid v(x_i) \in B(x_i)\}$ *and*

$$\mathcal{P}_{\mathsf{sub}}(t, \beta, t') = \begin{cases} 1 & \text{if } t \in S_\circ \text{ and } t' = \langle t, \beta\rangle \in S_\square, \\ \mathcal{P}(s, \alpha, t')[\beta] & \text{if } t = \langle s, \alpha\rangle \in S_\square, \beta \in Act_s^\alpha, \text{ and } t' \in S_\circ, \\ 0 & \text{otherwise.} \end{cases}$$

We now relate the obtained stochastic game $\mathcal{G} = \mathsf{sub}_r(\mathcal{M})$ under different schedulers for player $\circ$ with the substitution in the scheduler-induced pMCs of $\mathcal{M}$. We observe that the schedulers $\sigma \in \mathfrak{S}_\circ(\mathcal{G})$ for player $\circ$ coincide with the schedulers in $\mathcal{M}$. Consider $\mathcal{G}^\sigma$ with $\sigma \in \mathfrak{S}_\circ(\mathcal{G})$ which arises from $\mathcal{G}$ by erasing transitions not agreeing with σ, i. e., we set all $\mathcal{P}_\mathcal{G}(s, \alpha, \langle s, \alpha\rangle)$ with $s \in S_\circ$ and $\alpha \neq \sigma(s)$ to zero. Note that $\mathcal{G}^\sigma$ is an MDP as at each state of player $\circ$, only one action is enabled and therefore only player $\square$ has nondeterministic choices.

Example 9. Continuing Example 8, applying scheduler σ to $\mathcal{G}$ yields $\mathcal{G}^\sigma$, see Fig. 5(e). The MDP $\mathcal{G}^\sigma$ matches the MDP $\mathsf{sub}_r(\mathcal{M}^\sigma)$ apart from intermediate states of the form $\langle s, \alpha\rangle$: The state s in $\mathsf{sub}_r(\mathcal{M}^\sigma)$ has the same outgoing transitions as the state $\langle s, \alpha\rangle$ in $\mathcal{G}^\sigma$ and $\langle s, \alpha\rangle$ is the unique successor of s in $\mathcal{G}^\sigma$.

Note that $\mathcal{G}^\sigma$ and $\mathsf{sub}_r(\mathcal{M}^\sigma)$ induce the same reachability probabilities. Formally:

Corollary 3. *For pMDP $\mathcal{M}$, well-defined region r, target states $T \subseteq S$, and schedulers $\sigma \in \mathfrak{S}_{\circ}(\mathit{sub}_r(\mathcal{M}))$ and $\rho \in \mathfrak{S}(\mathit{sub}_r(\mathcal{M}^\sigma))$, it holds that*

$$\Pr^{(\mathit{sub}_r(\mathcal{M}^\sigma))^\rho}(\Diamond T) = \Pr^{\mathit{sub}_r(\mathcal{M})^{\sigma,\widehat{\rho}}}(\Diamond T)$$

with $\widehat{\rho} \in \mathfrak{S}_{\square}(\mathit{sub}_r(\mathcal{M}))$ satisfies $\widehat{\rho}(\langle s, \sigma(s)\rangle) = \rho(s)$.

Instead of performing the substitution on the pMC induced by $\mathcal{M}$ and σ, we can perform the substitution on $\mathcal{M}$ directly and preserve the reachability probability.

Theorem 3. *Let $\mathcal{M}$ be a pMDP, r be a well-defined region. Then*

$$\mathit{sub}_r(\mathcal{M}) \models_{\emptyset} \mathbb{P}_{\leq\lambda}(\Diamond T) \text{ implies } \mathcal{M}, r \models \mathbb{P}_{\leq\lambda}(\Diamond T), \text{ and}$$
$$\mathit{sub}_r(\mathcal{M}) \models_{\{\circ\}} \mathbb{P}_{>\lambda}(\Diamond T) \text{ implies } \mathcal{M}, r \models \neg\mathbb{P}_{\leq\lambda}(\Diamond T).$$

Therefore, analogously to the pMC case (cf. Theorem 2), we can derive whether $\mathsf{sub}_r(\mathcal{M}) \models \varphi$ by analysing a stochastic game. The formal proof is in [20].

5 Parameter Synthesis

In this section we briefly discuss how the regional model checking is embedded into a complete parameter space partitioning framework as, e. g., described in [12]. The goal is to partition the parameter space into *safe* and *unsafe* regions (cf. Sect. 3.1). From a practical point of view, yielding a 100 % coverage of the parameter space is not realistic; instead a large coverage (say, 95 %) is aimed at.

We discuss the complete chain for a pMDP $\mathcal{M}$ and a property φ. In addition, a well-defined region R is given which serves as *parameter space*. Recall that a region $r \subseteq R$ is safe or unsafe if $\mathcal{M}, r \models \varphi$ or $\mathcal{M}, r \models \neg\varphi$, respectively. Note that parameter space partitioning is also applicable if only parts of R are well-defined, as well-definedness of a region is effectively decidable and such (sub-)regions can simply be tagged as *not defined* and treated as being inconclusive.

As a *preprocessing* step, the input model is simplified by reducing its state space. First, *bisimulation minimisation* for parametric probabilistic models [15] is used. Then, *state elimination* [13] is applied to all states with $V_s^\alpha = \emptyset$ and $|Act(s)| = 1$. We then construct the parameter-substitution of the model. As the topology of the substitution is independent of the region, for checking multiple regions we simply substitute the probabilities according to the region of interest.

Now, using a heuristic from the parameter space partitioning framework, we determine a candidate region. A naive heuristic would be to start with R, and then to split inconclusive regions along each dimension recursively – as in [10], thereby reducing the over-approximation. More evolved heuristics apply some instantiations of the model to construct candidate regions [12].

For a candidate region $r \subseteq R$, regional model checking (Sects. 3 and 4) determines it to be safe or unsafe. Moreover, the result for a region may be *inconclusive*, which might occur if r is neither safe nor unsafe, but also if the approximation was too coarse. The procedure stops as soon as a sufficiently large area of the parameter space R has been classified into safe and unsafe regions.

6 Experimental Evaluation

We implemented and analysed the *parameter lifting algorithm* (PLA) as described in Sects. 3 and 4. Moreover, we connected the implementation with the parameter synthesis framework `PROPhESY` [12].

Setup. We implemented PLA in C++. Solving the resulting non-parametric systems is done via value iteration (using sparse matrices) with a precision of $\varepsilon = 10^{-6}$. We evaluated the performance and compared it to parameter space partitioning in `PARAM` and in `PRISM`, both based on [10] and using an unsound heuristic in the implementation. The experiments were conducted on an HP BL685C G7, 48 cores, 2.0 GHz each, and 192 GB of RAM. We restricted the RAM to 30 GB and set a time-out of one hour for all experiments. Our PLA implementation used a single core only. We consider the well-known pMC and pMDP benchmarks from [16]. We additionally translated existing MDPs for a semi-autonomous vehicle [21] and the zeroconf protocol [22] into pMDPs, cf. [20]. For each instance, we analysed the parameter space $R = [10^{-5}, 1-10^{-5}]^{\#\text{pars}}$ until 95 % is classified as safe or unsafe. Regions for which no decisive result was found were split into equally large regions, thus mimicking the behaviour of [10]. We also compared PLA to the SMT-based synthesis for pMCs in [12]. However, using naive heuristics for determining region candidates, the SMT solver often spent too much time for checking certain regions. For the desired coverage of 95 %, this led to timeouts for all tested benchmarks.

Results. The results are summarised in Table 1, listing the benchmark set and the particular instance. Further columns reflect whether a reachability or an expected reward property was checked ($\mathbb{P}$ vs. $\mathbb{E}$) and the number of *parameters*, *states* and *transitions*. We used the properties as given in their sources, for details see [20]. We ran PLA in two different settings: With strong bisimulation minimisation (*bisim*) and without *(direct)*. We list the number of considered *regions*, i. e., those required to cover >95 % of the parameter space, and the required run time in seconds for the complete verification task, including model building and preprocessing. For `PRISM`, we give the fastest run time producing a correct result out of 30 different possible configurations, differing in the performed bisimulation minimisation (none, strong, weak), how inconclusive regions are split (each or longest edge), and the order of states (all except "random"). The `PRISM` implementation was superior to the `PARAM` implementation in all cases. The sound variant of `PRISM` and `PARAM` would require SMT calls similar to [12], decreasing their performance.

To evaluate the approximation quality, we additionally ran PLA for 625 equally large regions that were not refined in the case of indecisive results. We depict detailed results for a selection in Table 2, where we denote model, instance, property type, number of parameters, states and transitions as in Table 2. Column *#par trans* lists the number of transitions labeled with a non-constant function. Running times are given in column t. Next, we show the percentage of regions that our approach could conclusively identify as safe or unsafe. For the remaining regions, we sampled the model at the corner points to analyse the

Table 1. Runtimes of synthesis on different benchmark models.

							PLA			PRISM
	Benchmark	Instance	φ	#Pars	#States	#Trans	#Regions	Direct	Bisim	Best
pMC	brp	(256,5)	$\mathbb{P}$	2	19 720	26 627	37	**6**	14	TO
		(4096,5)	$\mathbb{P}$	2	315 400	425 987	13	**233**	TO	TO
		(256,5)	$\mathbb{E}$	2	20 744	27 651	195	**8**	15	TO
		(4096,5)	$\mathbb{E}$	2	331 784	442 371	195	502	**417**	TO
		(16,5)	$\mathbb{E}$	4	1 304	1 731	1 251 220	2 764	**1 597**	TO
		(32,5)	$\mathbb{E}$	4	2 600	3 459	1 031 893	TO	**2 722**	TO
		(256,5)	$\mathbb{E}$	4	20 744	27 651	–	TO	TO	TO
	crowds	(10,5)	$\mathbb{P}$	2	104 512	246 082	123	17	**6**	2038
		(15,7)	$\mathbb{P}$	2	8 364 409	25 108 729	116	1 880	**518**	TO
		(20,7)	$\mathbb{P}$	2	45 421 597	164 432 797	119	TO	**2 935**	TO
	nand	(10,5)	$\mathbb{P}$	2	35 112	52 647	469	**22**	30	TO[a]
		(25,5)	$\mathbb{P}$	2	865 592	1 347 047	360	**735**	2 061	TO
pMDP	brp	(256,5)	$\mathbb{P}$	2	40 721	55 143	37	**35**	3 359	TO
		(4096,5)	$\mathbb{P}$	2	647 441	876 903	13	**3 424**	TO	TO
	consensus	(2,2)	$\mathbb{P}$	2	272	492	119	**<1**	**<1**	31[a]
		(2,32)	$\mathbb{P}$	2	4 112	7 692	108	**113**	141	TO[a]
		(4,2)	$\mathbb{P}$	4	22 656	75 232	6 125	**1 866**	2 022	TO[a]
		(4,4)	$\mathbb{P}$	4	43 136	144 352	–	TO	TO	TO[a]
	sav	(6,2,2)	$\mathbb{P}$	2	379	1 127	162	**<1**	**<1**	TO[a]
		(100,10,10)	$\mathbb{P}$	2	1 307 395	6 474 535	37	**1 612**	TO	TO
		(6,2,2)	$\mathbb{P}$	4	379	1 127	621 175	944	**917**	TO[a]
		(10,3,3)	$\mathbb{P}$	4	1 850	6 561		TO	TO	TO[a]
	zeroconf	(2)	$\mathbb{P}$	2	88 858	203 550	186	**86**	1 295	TO
		(5)	$\mathbb{P}$	2	494 930	1 133 781	403	**2 400**	TO	TO

[a]The fastest PRISM configuration gave an incorrect answer

Table 2. Results for classification of a constant number of regions.

		Instance	φ	#Pars	#States	#Trans	#Par trans	t	Safe	Unsafe	Neither	Unkn
pMC	Brp	(256,5)	$\mathbb{E}$	2	20 744	27 651	13 814	**51**	14.9 %	79.2 %	**5.8%**	**0.2%**
		(256,5)	$\mathbb{E}$	4	20 744	27 651	13 814	**71**	7.5 %	51.0 %	**40.6%**	**0.8%**
	Crowds	(10,5)	$\mathbb{P}$	2	104 512	246 082	51 480	**44**	54.4 %	41.1 %	**4.2%**	**0.3%**
	Nand	(10,5)	$\mathbb{P}$	2	35 112	52 647	25 370	**21**	21.4 %	68.5 %	**6.9%**	**3.2%**
pMDP	brp	(256,5)	$\mathbb{P}$	2	40 721	55 143	27 800	**153**	6.6 %	90.4 %	**3.0%**	**0.0%**
	Consensus	(4,2)	$\mathbb{P}$	4	22 656	75 232	29 376	**357**	2.6 %	87.0 %	**10.4%**	**0.0%**
	Aav	(6,2,2)	$\mathbb{P}$	4	379	1 127	552	**2**	44.0 %	15.4 %	**35.4%**	**5.3%**
	Zeroconf	(2)	$\mathbb{P}$	2	88 858	203 550	80 088	**186**	16.6 %	77.3 %	**5.6%**	**0.5%**

approximation error. Column *neither* gives the percentage of regions for which the property is neither always satisfied, nor always violated (as obtained from the sampling). In these cases, the inconclusive result is not caused by the approximation error but by the region selection. Finally, the fraction of the remaining regions for which it is still unknown if they are safe, unsafe or neither is given in column *unkn*.

Observations. PLA outperforms existing approaches by several orders of magnitude. We see two major reasons. First, the approach exploits the structure of parametric models, in which transition probabilities are usually described by

simple functions. This is a major benefit over state-elimination based approaches where any structure is lost. Secondly, the approach benefits from the speed of the numerical approaches used in non-parametric probabilistic verification. However, it is well known that problems due to numerical instability are an issue here. Furthermore, when checking a single region, the number of parameters has only a minor influence on the runtime; more important is the number of states and the graph-structure. However, the number of required regions grows exponentially in the number of parameters. Therefore, investigating good heuristics for the selection of candidate regions proves to be essential. Nevertheless, already the naive version used here yields a superior performance.

Table 2 shows that the over-approximation of PLA is sufficiently tight to immediately cover large parts of the parameter space. In particular, for all benchmark models with two parameters, we can categorise more than 89 % of the parameter space as safe/unsafe within less than four minutes. For four parameters, we cannot cover as much space due to the poor choice of regions: A lot of regions cannot be proven (un)safe, because they are in fact neither (completely) safe nor unsafe and not because of the approximation. This is tightly linked with the observed increase in runtime for models with four parameters in Table 1 since it implies that regions have to be split considerably before a decision can be made. The minimal number of regions depends only on the property and the threshold used, as in [10] and in [12]. PLA might need additional regions (although empirically, this is not significant), this corresponds to the practical case in [12] when regions are split just due to a time-out of the SMT-solver.

7 Conclusion

This paper presented parameter lifting, a new approach for parameter synthesis of Markov models. It relies on replacing parameters by nondeterminism, scales well, and naturally extends to treating parametric MDPs.

Acknowledgement. This work was supported by the Excellence Initiative of the German federal and state government, and the CDZ project CAP (GZ 1023).

References

1. 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). doi:10.1007/978-3-642-22110-1_47
2. Hahn, E.M., Li, Y., Schewe, S., Turrini, A., Zhang, L.: iscasMc: a web-based probabilistic model checker. In: Jones, C., Pihlajasaari, P., Sun, J. (eds.) FM 2014. LNCS, vol. 8442, pp. 312–317. Springer, Heidelberg (2014). doi:10.1007/978-3-319-06410-9_22
3. Chen, T., Forejt, V., Kwiatkowska, M., Parker, D., Simaitis, A.: PRISM-games: a model checker for stochastic multi-player games. In: Piterman, N., Smolka, S.A. (eds.) TACAS 2013. LNCS, vol. 7795, pp. 185–191. Springer, Heidelberg (2013). doi:10.1007/978-3-642-36742-7_13

4. 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). doi:10.1007/978-3-642-19835-9_30
5. Calinescu, R., Ghezzi, C., Johnson, K., Pezze, M., Rafiq, Y., Tamburrelli, G.: Formal verification with confidence intervals: a new approach to establishing the quality-of-service properties of software systems. IEEE Trans. Rel. **65**(1), 107–125 (2016)
6. Češka, M., Dannenberg, F., Kwiatkowska, M., Paoletti, N.: Precise parameter synthesis for stochastic biochemical systems. In: Mendes, P., Dada, J.O., Smallbone, K. (eds.) CMSB 2014. LNCS, vol. 8859, pp. 86–98. Springer, Heidelberg (2014). doi:10.1007/978-3-319-12982-2_7
7. Long, F., Rinard, M.: Automatic patch generation by learning correct code. In: Bodik, R., Majumdar, R., eds. POPL, pp. 298–312. ACM (2016)
8. Su, G., Rosenblum, D.S.: Nested reachability approximation for discrete-time Markov chains with univariate parameters. In: Cassez, F., Raskin, J.-F. (eds.) ATVA 2014. LNCS, vol. 8837, pp. 364–379. Springer, Heidelberg (2014). doi:10.1007/978-3-319-11936-6_26
9. Su, G., Rosenblum, D.S., Tamburrelli, G.: Reliability of run-time quality-of-service evaluation using parametric model checking. In: ICSE. ACM (2016, to appear)
10. 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). doi:10.1007/978-3-642-20398-5_12
11. Hahn, E.M., Hermanns, H., Wachter, B., Zhang, L.: PARAM: a model checker for parametric Markov models. In: Touili, T., Cook, B., Jackson, P. (eds.) CAV 2010. LNCS, vol. 6174, pp. 660–664. Springer, Heidelberg (2010). doi:10.1007/978-3-642-14295-6_56
12. Dehnert, C., Junges, S., Jansen, N., Corzilius, F., Volk, M., Bruintjes, H., Katoen, J.P., Ábrahám, E.: PROPhESY: a probabilistic parameter synthesis tool. In: Kroening, D., Păsăreanu, C.S. (eds.) CAV. LNCS, vol. 9206, pp. 214–231. Springer, Heidelberg (2015)
13. 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). doi:10.1007/978-3-540-31862-0_21
14. Brim, L., Češka, M., Dražan, S., Šafránek, D.: Exploring parameter space of stochastic biochemical systems using quantitative model checking. In: Sharygina, N., Veith, H. (eds.) CAV 2013. LNCS, vol. 8044, pp. 107–123. Springer, Heidelberg (2013). doi:10.1007/978-3-642-39799-8_7
15. Hahn, E.M., Hermanns, H., Zhang, L.: Probabilistic reachability for parametric Markov models. STTT **13**(1), 3–19 (2010)
16. PARAM Website: (2015). http://depend.cs.uni-sb.de/tools/param/
17. Vardi, M.Y.: Automatic verification of probabilistic concurrent finite-state programs. In: FOCS, pp. 327–338. IEEE CS (1985)
18. Baier, C., Katoen, J.P.: Principles of Model Checking. The MIT Press, Cambridge (2008)
19. Puterman, M.L.: Markov Decision Processes: Discrete Stochastic Dynamic Programming. Wiley, Hoboken (1994)
20. Quatmann, T., Dehnert, C., Jansen, N., Junges, S., Katoen, J.P.: Parameter synthesis for Markov models: faster than ever. CoRR abs/1602.05113 (2016)

21. Junges, S., Jansen, N., Dehnert, C., Topcu, U., Katoen, J.-P.: Safety-constrained reinforcement learning for MDPs. In: Chechik, M., Raskin, J.-F. (eds.) TACAS 2016. LNCS, vol. 9636, pp. 130–146. Springer, Heidelberg (2016). doi:10.1007/978-3-662-49674-9_8
22. Kwiatkowska, M., Norman, G., Parker, D., Sproston, J.: Performance analysis of probabilistic timed automata using digital clocks. FMSD **29**, 33–78 (2006)