

Probabilistic Bisimulations for PCTL Model Checking of Interval MDPs

Vahid Hashemi

MPI für Informatik
Saarbrücken, Germany
Department of Computer Science
Saarland University
Saarbrücken, Germany
hashemi@mpi-inf.mpg.de

Hassan Hatefi

Department of Computer Science
Saarland University
Saarbrücken, Germany
hhatefi@cs.uni-saarland.de

Jan Krčál

Department of Computer Science
Saarland University
Saarbrücken, Germany
krcal@cs.uni-saarland.de

Verification of PCTL properties of MDPs with convex uncertainties has been investigated recently by Puggelli et al. However, model checking algorithms typically suffer from state space explosion. In this paper, we address probabilistic bisimulation to reduce the size of such an MDPs while preserving PCTL properties it satisfies. We discuss different interpretations of uncertainty in the models which are studied in the literature and that result in two different definitions of bisimulations. We give algorithms to compute the quotients of these bisimulations in time polynomial in the size of the model and exponential in the uncertain branching. Finally, we show by a case study that large models in practice can have small branching and that a substantial state space reduction can be achieved by our approach.

1 Introduction

Modelling formalisms like Markov decision processes (MDP) [29] or Probabilistic automata (PA) [31] are used for representing systems that combine non-deterministic and probabilistic behaviour. They can be viewed as transition systems where in each step an outgoing transition of the current state is chosen *non-deterministically* and the successor state is chosen *randomly* according to a fixed probability distribution assigned to this transition. Assigning fixed probability distributions to transitions is however not realistic [18,22] in many modelling scenarios: measurement errors, statistical estimates, or mathematical approximations all lead to *intervals* instead of fixed probabilities.

Interval MDPs [28] (also called *Bounded-parameter MDPs* [13, 37]) address this need by bounding the probabilities of each successor state by an interval instead of a fixed number. In such a model, the transition probabilities are not fully specified and this uncertainty again needs to be resolved non-deterministically. The two sources of non-determinism have *different* interpretation in different applications:

1. In verification of parallel systems with uncertain transition probabilities [28] the transitions correspond to unpredictable interleaving of computation of the communicating agents. Hence, both the choice of transitions and their probability distributions is *adversarial*.
2. In control synthesis for systems with uncertain probabilities [36] the transitions correspond to various control actions. We search for a choice of transitions that is *optimal* against an adversarial choice of probability distributions satisfying the interval bounds.
3. In parameter synthesis for parallel systems [14] the transition probabilities are underspecified to allow freedom in implementation of such a model. We search for a choice of probability dis-

tributions that is optimal for adversarial choice of transitions (again stemming from the possible interleaving).

Furthermore, the choice of probability distributions satisfying the interval constraints can be either resolved statically [18], i.e. at the beginning once for all, or dynamically [17, 33], i.e. independently for each computation step. Here, we focus on the dynamic approach that is easier to work with algorithmically and can be seen as a relaxation of the static approach that is often intractable [2, 7, 11, 33].

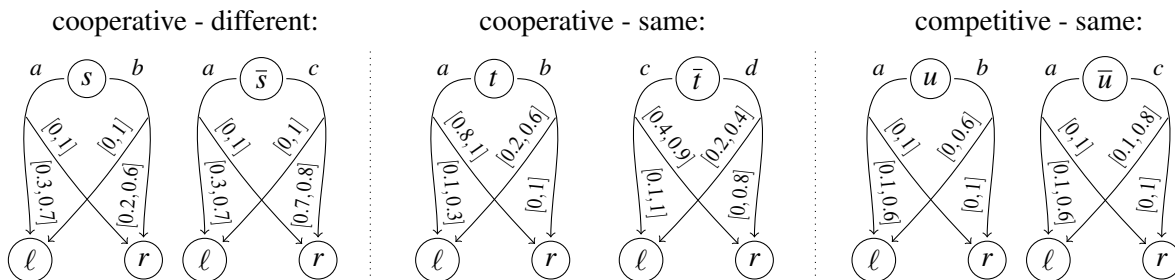
There are several algorithms [28, 36] to check whether a given interval MDP satisfies a given specification expressed in a *logic* like PCTL [15] or LTL [27]. However, models often suffer from state space explosion when obtained using some higher-level modelling formalism such as a process algebra. These models usually contain redundancy that can be removed without changing the behaviour of the model. One way to reason about such behavioural equivalence is *bisimulation* [24]. For a given huge model it allows to construct the *bisimulation quotient*, the smallest model with equivalent behaviour – in particular preserving all its properties expressible by a logic such as PCTL.

Our contribution In this paper, we define the first bisimulations for interval MDPs (that are also the first bisimulations for MDPs with uncertain transitions in general). We show that different interpretation of non-determinism yields two different bisimulations: one for models where the two non-determinisms are resolved in a *cooperative way* (see point 1. above), another for models where it is resolved in a *competitive way* (see points 2. and 3. above).

Furthermore, we show how to compute these bisimulations by algorithms based on comparing polytopes of probability distributions associated with each transition. The algorithms are fixed parameter tractable with respect to the maximal dimension of the polytopes (i.e. maximal number of different states that an uncertain transition can lead to); in the competitive case also with respect to the maximal number of outgoing uncertain transitions. Note that in many applications these parameters are small.

We finally argue by a case study that, if uncertainty stems from a small number of different phenomena such as *node failure* or *loss of a message*, the same shape of polytopes will repeat many times over the states space. We demonstrate that the redundancy in this case may result in a massive state space minimisation.

Example 1.1. We illustrate the contribution by two examples. In the first one, we explain how the competitive and the cooperative resolution of non-determinism result in different behavioural equivalences. Consider the three pair of states below.



As regards the cooperative non-determinism, s has not the same behaviour as \bar{s} since \bar{s} can move to r with probability 0.8 by choosing c and ($\ell \mapsto 0.2, r \mapsto 0.8$), which s cannot simulate. So far the equivalence might seem easy to check. However, note that t has the same behaviour as \bar{t} even though the interval bounds for the transitions quite differ. Indeed, the sets of distributions satisfying the interval constraints are the same for t and \bar{t} .

general polynomial constraints on probabilities instead of interval bounds. Since for Interval MC it is not possible to explicitly construct parallel composition, the problem whether there is a common implementation of a set of Interval Markov chains is addressed instead [11]. To the contrary, interval bounds on *rates* of outgoing transitions work well with parallel composition in the continuous-time setting of Abstract interactive Markov chains [20]. The reason is that unlike probabilities, the rates do not need to sum up to 1. A different way [38] to successfully define parallel composition for interval models is to separate synchronising transitions from the transitions with uncertain probabilities. This is also the core of our approach to parallel composition when constructing a case study as discussed in Section 5.

We are not aware of any existing bisimulation for uncertain or parametric probabilistic models. Among similar concepts studied in the literature are simulation [38] and refinement [9, 11, 18] relations for previously mentioned models. Our definition of bisimulation in the competitive setting is inspired by the alternating bisimulation [1, 6].

Many new verification algorithms for interval models appeared in last few years. Reachability and expected total reward is addressed for Interval MC [8] as well as Interval MDP [37]. PCTL model checking and LTL model checking are studied for Interval MC [2, 7, 8] and also for Interval MDP [28, 36]. Among other technical tools, all these approaches make use of (robust) dynamic programming relying on the fact that transition probability distributions are resolved dynamically. For the static resolution of distributions, adaptive discretisation technique for PCTL parameter synthesis is given in [14]. Uncertain models are also widely studied in the control community [13, 25, 37], mainly interested in maximal expected finite-horizon reward or maximal expected discounted reward.

Structure of the paper We start with necessary preliminaries in Section 2. In Section 3, we give the definitions of probabilistic bisimulations for interval MDP and discuss their properties and differences. In Section 4, we give the FPT algorithms for both cooperative and competitive cases. Finally, in Section 5 we demonstrate our approach on a case study. Due to space limitations, we refer the reader interested in detailed proofs to [16].

2 Preliminaries

In this paper, the sets of all positive integers, rational numbers, real numbers and non-negative real numbers are denoted by \mathbb{N} , \mathbb{Q} , \mathbb{R} , and $\mathbb{R}^{\geq 0}$, respectively. For a set X , we denote by $\Delta(X)$ the set of discrete probability distributions over X .

2.1 Interval Markov Decision Processes

Let us formally define Interval *MDP*.

Definition 1 (IMDP). An Interval Markov Decision Process (IMDP) \mathfrak{M} is a tuple (S, A, AP, L, I) , where S is a finite set of states, A is a finite set of actions, AP is a finite set of atomic propositions, $L: S \rightarrow 2^{AP}$ is a labelling function, and $I: S \times A \times S \rightarrow \mathbb{I}$ is an interval transition probability function where \mathbb{I} is a set of subintervals of $[0, 1]$.

Furthermore, for each state s and action a , we denote by $s \xrightarrow{a} \mu$ that $\mu \in \Delta(S)$ is a *feasible distribution*, i.e. for each state s' we have $\mu(s') \in I(s, a, s')$. We require that the set $\{\mu \mid s \xrightarrow{a} \mu\}$, also denoted by $\mathcal{E}^{s,a}$, is non-empty for each state s and action a .

An interval MDP is initiated in some state s_1 and then moves in discrete steps from state to state forming an infinite path $s_1 s_2 s_3 \dots$. One step, say from state s_i , is performed as follows. First, an action $a \in A$ is chosen non-deterministically by *Scheduler*. Then, *Nature* resolves the uncertainty and chooses non-deterministically one corresponding feasible distribution $\mu \in \mathcal{E}^{s_i, a}$. Finally, the next state s_{i+1} is chosen randomly according to the distribution μ .

Let us define the semantics of an *IMDP* formally. A *path* is a finite or infinite sequence of states $\omega = s_1 s_2 \dots$. For a finite path ω , we denote by $last(\omega)$ the last state of ω . The set of all finite paths and the set of all infinite paths are denoted by $Paths_{fin}$ and $Paths_{inf}$, respectively. Furthermore, let $Paths_\omega = \{\omega\omega' \mid \omega' \in Paths_{inf}\}$ denote the set of paths that have the finite prefix $\omega \in Paths_{fin}$.

Definition 2 (*Scheduler and Nature*). A scheduler is a function $\sigma : Paths_{fin} \rightarrow \Delta(A)$ that to each finite path ω assigns a distribution over the set of actions. A nature is a function $\pi : Paths_{fin} \times A \rightarrow \Delta(S)$ that to each finite path ω and action a assigns a feasible distribution, i.e. an element of $\mathcal{E}^{s, a}$ where $s = last(\omega)$. We denote by Σ the set of all schedulers and by Π the set of all natures.

For an initial state s , a scheduler σ , and a nature π , let $\Pr_s^{\sigma, \pi}$ denote the unique probability measure over $(Paths_{inf}, \mathcal{B})^1$ such that the probability $\Pr_s^{\sigma, \pi}[Paths_{s'}]$ of starting in s' equals 1 if $s = s'$ and 0, otherwise; and the probability $\Pr_s^{\sigma, \pi}[Paths_{\omega s'}]$ of traversing a finite path $\omega s'$ equals $\Pr_s^{\sigma, \pi}[Paths_\omega] \cdot \sum_{a \in A} \sigma(\omega)(a) \cdot \pi(\omega, a)(s')$.

Observe that the scheduler does not choose an action but a *distribution* over actions. It is well-known [31] that such randomisation brings more power in the context of bisimulations. To the contrary, nature is not allowed to randomise over the set of feasible distributions $\mathcal{E}^{s, a}$. This is in fact not necessary, since the set $\mathcal{E}^{s, a}$ is closed under convex combinations. Finally, a scheduler σ is said to be *deterministic* if $\sigma(\omega)(a) = 1$ for some action a for all finite paths ω .

2.2 Probabilistic Computation Tree Logic (PCTL)

There are various ways how to describe properties of interval *MDPs*. Here we focus on *probabilistic CTL* (PCTL) [15]. The syntax of PCTL state formulas φ and PCTL path formulas ψ is given by:

$$\begin{aligned} \varphi &:= true \mid x \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid P_{\bowtie p}(\psi) \\ \psi &:= X\varphi \mid \varphi_1 \cup \varphi_2 \mid \varphi_1 \cup^{\leq k} \varphi_2 \end{aligned}$$

where $x \in AP$, $p \in [0, 1]$ is a rational constant, $\bowtie \in \{\leq, <, \geq, >\}$, and $k \in \mathbb{N}$.

The satisfaction relation for PCTL formulae depends on the way how non-determinism is resolved for the probabilistic operator $P_{\bowtie p}(\psi)$. When quantifying both the non-determinisms universally, we define the satisfaction relation $s \models_{(\forall)} \varphi$ as follows: $s \models_{(\forall)} x$ if $x \in L(s)$; $s \models_{(\forall)} \neg\varphi$ if not $s \models_{(\forall)} \varphi$; $s \models_{(\forall)} \varphi_1 \wedge \varphi_2$ if both $s \models_{(\forall)} \varphi_1$ and $s \models_{(\forall)} \varphi_2$; and

$$s \models_{(\forall)} P_{\bowtie p}(\psi) \quad \text{if} \quad \forall \sigma \in \Sigma \quad \forall \pi \in \Pi : \quad \Pr_s^{\sigma, \pi} [\models_{(\forall)} \psi] \bowtie p. \quad (\forall)$$

where $\models_{(\forall)} \psi$ denotes the set of infinite paths $\{\omega \in Paths_{inf} \mid \omega \models_{(\forall)} \psi\}$ and the satisfaction relation

¹ Here, \mathcal{B} is the standard σ -algebra over $Paths_{inf}$ generated from the set of all cylinder sets $\{Paths_\omega \mid \omega \in Paths_{fin}\}$. The unique probability measure is obtained by the application of the by extension theorem (see, e.g. [3]).

$\omega \models_{(\forall)} \psi$ for an infinite path $\omega = s_1 s_2 \dots$ and a path formula ψ is given by:

$$\begin{aligned} \omega \models_{(\forall)} \neg \varphi & \quad \text{if } s_1 \not\models \varphi; \\ \omega \models_{(\forall)} \varphi_1 \text{U}^{\leq k} \varphi_2 & \quad \text{if there exists } i \leq k \text{ such that } s_i \models_{(\forall)} \varphi_2, \\ & \quad \text{and } s_j \models_{(\forall)} \varphi_1 \text{ for every } 0 \leq j < i; \\ \omega \models_{(\forall)} \varphi_1 \text{U} \varphi_2 & \quad \text{if there exists } k \in \mathbb{N} \text{ such that } \omega \models_{(\forall)} \varphi_1 \text{U}^{\leq k} \varphi_2. \end{aligned}$$

It is easy to show that the set $\models_{(\forall)} \psi$ is measurable for any path formula ψ , hence the definition is correct. We explain how the semantics differs for different resolution of non-determinism in the next section.

3 Probabilistic Bisimulations for Interval Markov decision processes

Let us fix an interval MDP (S, A, AP, L, I) . In this section, we define probabilistic bisimulations for different interpretations of Interval MDP. Namely the bisimulation $\sim_{(\forall)}$ for the cooperative setting and bisimulations $\sim_{(\exists\sigma\forall)}$ and $\sim_{(\exists\pi\forall)}$ for two different applications for the competitive setting. We then show that $\sim_{(\exists\sigma\forall)}$ and $\sim_{(\exists\pi\forall)}$ actually coincide.

3.1 Cooperative resolution of non-determinism

In the context of verification of parallel systems with uncertain transition probabilities, it makes sense to assume that *Scheduler* and *Nature* are resolved *cooperatively* in the most *adversarial* way. This setting yields a bisimulation quite similar to standard probabilistic bisimulation for models with one type of non-determinism [23]. First, let us denote by $s \longrightarrow \mu$ that a transition from s according to μ can be taken cooperatively, i.e. that there is a decision $\rho \in \text{Dist}(A)$ of *Scheduler* and decisions $s \xrightarrow{a} \mu_a$ of *Nature* for each a such that $\mu = \sum_{a \in A} \rho(a) \cdot \mu_a$. In other words, $s \longrightarrow \mu$ if $\mu \in \text{conv}\{\mathcal{E}^{s,a} \mid a \in A\}$ where $\text{conv}X$ denotes the convex hull of X .

Definition 3. Let $R \subseteq S \times S$ be an equivalence relation. We say that R is probabilistic (\forall) -bisimulation if for any $(s, t) \in R$ we have that $L(s) = L(t)$ and

$$\begin{aligned} & \text{for each } s \longrightarrow \mu \\ & \text{there is } t \longrightarrow \nu \text{ such that } \mu(\mathcal{C}) = \nu(\mathcal{C}) \text{ for each equivalence class } \mathcal{C} \in S/R. \end{aligned}$$

Furthermore, we write $s \sim_{(\forall)} t$ if there is a probabilistic (\forall) -bisimulation R such that $(s, t) \in R$.

Intuitively, each (cooperative) step of *Scheduler* and *Nature* from state s needs to be matched by a (cooperative) step of *Scheduler* and *Nature* from state t ; symmetrically, s also needs to match t . As a first result, we show that the bisimulation $\sim_{(\forall)}$ preserves the (cooperative) universally quantified PCTL satisfaction $\models_{(\forall)}$.

Theorem 1. For states $s \sim_{(\forall)} t$ and any PCTL formula φ , we have $s \models_{(\forall)} \varphi$ if and only if $t \models_{(\forall)} \varphi$.

Dually, the non-determinism could also be resolved *existentially*. This corresponds to the setting where we want to synthesise both the scheduler σ that controls the system and choice of feasible probability distributions π such that σ and π together guarantee a specified behaviour φ . This setting is formalised by the satisfaction relation $\models_{(\exists)}$ which is defined like $\models_{(\forall)}$ except for the operator $P_{\bowtie p}(\psi)$ where we set

$$s \models_{(\exists)} P_{\bowtie p}(\psi) \quad \text{if } \exists \sigma \in \Sigma \exists \pi \in \Pi: \Pr_s^{\sigma, \pi} [\models_{(\exists)} \psi] \bowtie p. \quad (\exists)$$

Note that for any formula of the form $P_{<p}(\psi)$, we have $s \models_{\exists} P_{<p}(\psi)$ if and only if we have $s \models_{(\forall)} \neg P_{\geq p}(\psi)$. This can be easily generalised: for each state formula φ we obtain a state formula $\bar{\varphi}$ such that $s \models_{\exists} \varphi$ if and only if $s \models_{(\forall)} \bar{\varphi}$ for each state s . Hence $\sim_{(\forall)}$ also preserves \models_{\exists} .

Corollary 1. *For states $s \sim_{(\forall)} t$ and any PCTL formula φ , we have $s \models_{(\exists)} \varphi$ if and only if $t \models_{\exists} \varphi$.*

3.2 Competitive resolution of non-determinism

As already argued for in Section 1, there are applications where it is natural to interpret the two sources of non-determinism in a competitive way.

Control synthesis under uncertainty In this setting we search for a scheduler σ such that for any nature π , a fixed property φ is satisfied. This corresponds to the satisfaction relation $\models_{(\exists\sigma\forall)}$, obtained similarly from $\models_{(\forall)}$ by replacing the rule (\forall) with

$$s \models_{(\exists\sigma\forall)} P_{\bowtie p}(\psi) \quad \text{if} \quad \exists \sigma \in \Sigma \quad \forall \pi \in \Pi : \quad \text{Pr}_s^{\sigma, \pi} [\models_{(\exists\sigma\forall)} \psi] \bowtie p. \quad (\exists\sigma\forall)$$

As regards bisimulation, the competitive setting is not a common one. We define a bisimulation similar to the alternating bisimulation of [1] applied to non-stochastic two-player games. For a decision $\rho \in \Delta(A)$ of *Scheduler*, let us denote by $s \xrightarrow{\rho} \mu$ that μ is a possible successor distribution, i.e. there are decisions μ_a of *Nature* for each a such that $\mu = \sum_{a \in A} \rho(a) \cdot \mu_a$.

Definition 4. *Let $R \subseteq S \times S$ be an equivalence relation. We say that R is probabilistic $(\exists\sigma\forall)$ -bisimulation if for any $(s, t) \in R$ we have that $L(s) = L(t)$ and*

$$\begin{aligned} & \text{for each } \rho_s \in \Delta(A) \\ & \text{there is } \rho_t \in \Delta(A) \\ & \text{such that for each } t \xrightarrow{\rho_t} \nu \\ & \text{there is } s \xrightarrow{\rho_s} \mu \text{ such that } \mu(\mathcal{C}) = \nu(\mathcal{C}) \text{ for each equivalence class } \mathcal{C} \in S/R. \end{aligned}$$

Furthermore, we write $s \sim_{(\exists\sigma\forall)} t$ if there is a probabilistic $(\exists\sigma\forall)$ -bisimulation R such that $(s, t) \in R$.

The exact alternation of quantifiers might be counter-intuitive at first sight. Note that it exactly corresponds to the situation in non-stochastic games [1] and that this bisimulation preserves the PCTL logic with $\models_{(\exists\sigma\forall)}$.

Theorem 2. *For states $s \sim_{(\exists\sigma\forall)} t$ and any PCTL formula φ , we have $s \models_{(\exists\sigma\forall)} \varphi$ if and only if $t \models_{(\exists\sigma\forall)} \varphi$.*

Similarly to Corollary 1, we could define a satisfaction relation with the alternation $\forall \sigma \in \Sigma \exists \pi \in \Pi$ that is then preserved by the same bisimulation $\sim_{(\exists\sigma\forall)}$. However, we see no natural application thereof.

Parameter synthesis in parallel systems In this setting, we search for a resolution π of the underspecified probabilities such that for any scheduler σ resolving the interleaving non-determinism, a fixed property φ is satisfied. This corresponds to the satisfaction relation $\models_{(\exists\pi\forall)}$, obtained similarly from $\models_{(\forall)}$ by replacing the rule (\forall) with

$$s \models_{(\exists\pi\forall)} P_{\bowtie p}(\psi) \quad \text{if} \quad \exists \pi \in \Pi \quad \forall \sigma \in \Sigma : \quad \text{Pr}_s^{\sigma, \pi} [\models_{(\exists\pi\forall)} \psi] \bowtie p. \quad (\exists\pi\forall)$$

This yields a definition of bisimulation similar to Definition 4. For a choice $(\mu_a)_{a \in A}$ of underspecified probabilities, let us denote by $s \xrightarrow{(\mu_a)} \mu$ that μ is a possible successor distribution, i.e. there is a decision ρ of *Scheduler* such that $\mu = \sum_{a \in A} \rho(a) \cdot \mu_a$.

Definition 5. Let $R \subseteq S \times S$ be a symmetric relation. We say that R is probabilistic $(\exists\pi\forall)$ -bisimulation if for any $(s, t) \in R$ we have that $L(s) = L(t)$ and

$$\begin{aligned} & \text{for each } (\mu_a)_{a \in A} \text{ where } s \xrightarrow{a} \mu_a \text{ for each } a \in A \\ & \text{there is } (\nu_a)_{a \in A} \text{ where } t \xrightarrow{a} \nu_a \text{ for each } a \in A \\ & \text{such that for each } t \xrightarrow{(\nu_a)} \nu \\ & \text{there is } s \xrightarrow{(\mu_a)} \mu \text{ such that } \mu(\mathcal{C}) = \nu(\mathcal{C}) \text{ for each equivalence class } \mathcal{C} \in S/R, \end{aligned}$$

Furthermore, we write $s \sim_{(\exists\pi\forall)} t$ if there is a probabilistic $(\exists\pi\forall)$ -bisimulation R such that $(s, t) \in R$.

The fact that this bisimulation preserves $\models_{(\exists\pi\forall)}$ can be proven analogously to Theorem 2.

Theorem 3. For states $s \sim_{(\exists\pi\forall)} t$ and any PCTL formula φ , we have $s \models_{(\exists\pi\forall)} \varphi$ if and only if $t \models_{(\exists\pi\forall)} \varphi$.

As a final result of this section, we show that these two bisimulations coincide.

Theorem 4. We have $\sim_{(\exists\sigma\forall)} = \sim_{(\exists\pi\forall)}$.

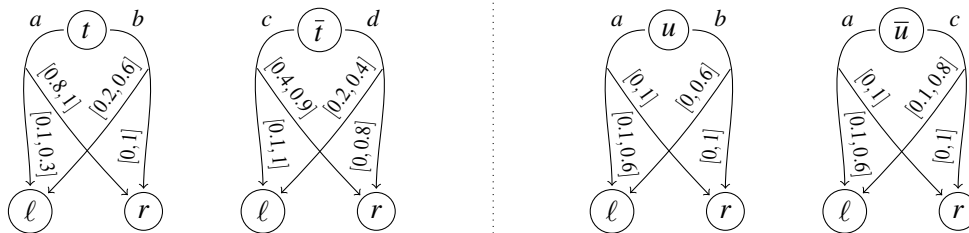
Thanks to this result, we denote from now on these coinciding bisimulations by $\sim_{(\exists\forall)}$. As a concluding remark, note that Definitions 3, 4 and 5 can be seen as the conservative extension of probabilistic bisimulation for (state-labelled) MDPs. To see that assume the set of uncertainty for every transition is a singleton. Since there is only one choice for the nature, the role of nature can be safely removed from the definitions. Moreover, it is worthwhile to note that Theorems 1, 2 and 3 show the soundness of the probabilistic bisimulation definitions with respect to PCTL. Unfortunately, it is shown in [31, 32] that probabilistic bisimulation for probabilistic automata is finer than PCTL equivalence which leads to the incompleteness in general. Since MDPs can be seen as a subclass of PAs, it is not difficult to see that the incompleteness holds also for MDPs.

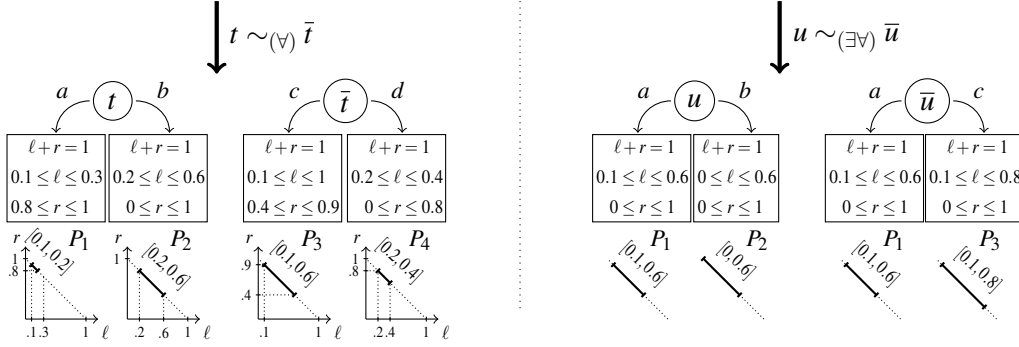
We also remark that the notions $\sim_{(\forall)}$ and $\sim_{(\exists\forall)}$ are incomparable, as it is for instance observable in Example 1.1. It is shown in the example that $t \sim_{(\forall)} \bar{t}$ and $u \sim_{(\exists\forall)} \bar{u}$. However it is not hard to verify that $t \not\sim_{(\exists\forall)} \bar{t}$ and $u \not\sim_{(\forall)} \bar{u}$. For the latter, notice that for example u can evolve to r with probability one by taking action b , whereas \bar{u} cannot simulate. The former is noticeable in the situation where the controller wants to maximise the probability to reach r , but the nature declines. In this case t chooses action b and the nature let it go to r with probability 0.8. Nevertheless the nature can prevent \bar{t} to evolve into r with probability more than 0.6, despite the fact which action has been chosen by \bar{t} .

4 Algorithms

In this section, we give algorithms for computing bisimulations $\sim_{(\forall)}$ and $\sim_{(\exists\forall)}$. We show that computing bisimulations in both cases is fixed-parameter tractable.

Example 4.1. Let us start by illustrating the ideas on Example 1.1 from Section 1.





The general sketch of the algorithm is as follows. We need to construct the polytopes of probability distributions offered by the actions; in our examples the polytopes are just line segments in two-dimensional space. We get $t \sim_{(\forall)} \bar{t}$ since the *convex hull* of P_1 and P_2 equals to the *convex hull* of P_3 and P_4 . Similarly, we get $u \sim_{(\exists\forall)} \bar{u}$ since u and \bar{u} have the same set of *minimal* polytopes w.r.t. set inclusion.

Let us state the results formally. Let us fix $\mathfrak{M} = (S, A, AP, L, I)$ where b is the maximal number of different actions $\max_{s \in S} |\{I(s, a, \cdot) \mid a \in A\}|$, f is the maximal support of an action $\max_{s \in S, a \in A} |\{s' \mid I(s, a, s') \neq [0, 0]\}|$, and $|\mathfrak{M}|$ denotes the size of the representation using adjacency lists for non-zero elements of I where we assume that the interval bounds are rational numbers encoded symbolically in binary.

Theorem 5. *There is an algorithm that computes $\sim_{(\forall)}$ in time polynomial in $|\mathfrak{M}|$ and exponential in f . There is also an algorithm that computes $\sim_{(\exists\forall)}$ in time polynomial in $|\mathfrak{M}|$ and exponential in f and b .*

Computing both bisimulations follows the standard partition refinement approach [4, 19, 26], formalized by the procedure `Bisimulation` in Algorithm 1. Namely, we start with R being the complete relation and iteratively remove from R pairs of states that violate the definition of bisimulation with respect to R . The core part is finding out whether two states “violate the definition of bisimulation”. This is where the algorithms for the two bisimulations differ.

4.1 Cooperative resolution of non-determinism of the bisimulation $\sim_{(\forall)}$

Let us first address $\sim_{(\forall)}$ where the violation is checked by the procedure `Violate(forall)`. We show that this amounts to checking inclusion of polytopes defined as follows. Recall that for $s \in S$ and an action $a \in A$, $\mathcal{E}^{s,a}$ denotes the polytope of feasible successor distributions over *states* with respect to taking the action a in the state s . By $\mathcal{E}_R^{s,a}$, we denote the polytope of feasible successor distributions over *equivalence classes* of R with respect to taking the action a in the state s . Formally, for $\mu \in \Delta(S/R)$ we set $\mu \in \mathcal{E}_R^{s,a}$ if we have

$$\mu(\mathcal{C}) \in \left[\sum_{s' \in \mathcal{C}} \inf I(s, a, s') \sum_{s' \in \mathcal{C}} \sup I(s, a, s') \right] \quad \text{for each } \mathcal{C} \in S/R.$$

Note that we require that the probability of each class \mathcal{C} must be in the interval of the sum of probabilities that can be assigned to states of \mathcal{C} . Furthermore, we define \mathcal{E}_R^s as the convex hull of $\bigcup_{a \in A} \mathcal{E}_R^{s,a}$. It is the set of feasible successor distributions over S/R with respect to taking an *arbitrary* distribution over actions in state s . As specified in the procedure `Violate(forall)`, we show that it suffices to check equality of these polytopes.

Proposition 1. *We have $s \sim_{(\forall)} t$ if and only if $L(s) = L(t)$ and $\mathcal{E}_{\sim_{(\forall)}}^s = \mathcal{E}_{\sim_{(\forall)}}^t$.*

we have $\mathcal{E}_R^{s,\rho} \subseteq \mathcal{E}_R^{s,a}$ where $\mathcal{E}_R^{s,\rho}$ denotes the polytope $\{\sum_{b \in A \setminus \{a\}} \rho(b) \cdot \mathbf{x}_b \mid \text{each } \mathbf{x}_b \in \mathcal{E}_R^{s,b}\}$.

Proposition 2. *We have $s \sim_{(\exists \forall)} t$ if and only if $L(s) = L(t)$ and $\{\mathcal{E}_{\sim(\exists \forall)}^{s,a} \mid a \in A, \mathcal{E}_{\sim(\exists \forall)}^{s,a} \text{ is strictly minimal}\} = \{\mathcal{E}_{\sim(\exists \forall)}^{t,a} \mid a \in A, \mathcal{E}_{\sim(\exists \forall)}^{t,a} \text{ is strictly minimal}\}$.*

Proof. We first address the “if” part. For each choice of *Nature* $(\mu_a)_{a \in A}$ where each $s \xrightarrow{a} \mu_a$, let $M = \{\bar{\mu}_a \mid a \in A\}$ and $M' \subseteq M$ be the subset where each distribution lies within some strictly minimal polytope $\mathcal{E}_{\sim(\exists \forall)}^{s,b}$. Because the strictly minimal polytopes coincide, we can construct a choice of *Nature* $N = (v_a)_{a \in A}$ such that $N = \{\bar{v}_a \mid a \in A\} = M'$. Because $N \subseteq M$, it is easy to see that for each $t \xrightarrow{(v_a)} v$ there is $s \xrightarrow{(\mu_a)} \mu$ such that $\mu(\mathcal{C}) = v(\mathcal{C})$ for each $\mathcal{C} \in S/R$.

As regards the “only if” part, let us assume that there is, say in t , a strictly minimal polytope $\mathcal{E}_{\sim(\exists \forall)}^{t,b}$ that is not in the set of strictly minimal polytopes for s . There is a choice of *Nature* $(\mu_a)_{a \in A}$ for state s such that no convex combination of elements of $M = \{\bar{\mu}_a \mid a \in A\}$ lies in $\mathcal{E}_{\sim(\exists \forall)}^{t,b}$; in particular no element of M lies in $\mathcal{E}_{\sim(\exists \forall)}^{t,b}$. For any choice of *Nature* $(v_a)_{a \in A}$ for state t , \bar{v}_b is not a convex combination of elements from M . Thus, if *Scheduler* chooses action b , there is no $s \xrightarrow{(\mu_a)} \mu$ such that $\mu(\mathcal{C}) = v_b(\mathcal{C})$ for each $\mathcal{C} \in S/R$ and it does *not* hold $s \sim_{(\exists \forall)} t$. \square

Next, we need to address how to compute whether a polytope is strictly minimal. We construct B and d such that $B\rho + d \geq 0$ implies $\mathcal{E}_R^{s,\rho} \subseteq \mathcal{E}_R^{s,a}$. Checking of strictly minimality then reduces to checking feasibility of this linear system. The system gets constructed iteratively. Let P_1, \dots, P_k denote the polytopes corresponding to all actions in s except for a . We enumerate all combinations $(c_1, \dots, c_k) \in C(P_1) \times \dots \times C(P_k)$ of corners of the polytopes. For each such combination we add into B and d new constraints $B_{(c_1, \dots, c_k)}$ and $d_{(c_1, \dots, c_k)}$ such that for any ρ satisfying $B_{(c_1, \dots, c_k)}\rho + d_{(c_1, \dots, c_k)} \geq 0$ we have $\sum \rho_i c_i \in \mathcal{E}_R^{s,a}$. For details, see the procedure `Minimal` in Algorithm 1.

Proposition 3. *We have $B\rho + d \geq 0$ is not feasible if and only if $\mathcal{E}_R^{s,a}$ is strictly minimal where the rows of B and d are obtained as a union of rows*

$$\begin{aligned} B &= \{\mathbf{1}, -\mathbf{1}\} \cup \bigcup \{B_{(c_1, \dots, c_k)} \mid (c_1, \dots, c_k) \in C(P_1) \times \dots \times C(P_k)\} \\ d &= \{-1, 1\} \cup \bigcup \{d_{(c_1, \dots, c_k)} \mid (c_1, \dots, c_k) \in C(P_1) \times \dots \times C(P_k)\}. \end{aligned}$$

Proof. Let ρ be any feasible solution of the system. It is easy to see that $\mathcal{E}_R^{s,\rho} \subseteq \mathcal{E}_R^{s,a}$ since $\mathcal{E}_R^{s,\rho}$ is convex and since all corners of $\mathcal{E}_R^{s,\rho}$ (obtained as a convex ρ -combination of corners of all $\mathcal{E}_R^{s,b}$) lie within $\mathcal{E}_R^{s,a}$. Hence, $\mathcal{E}_R^{s,a}$ is not strictly minimal. As regards the other direction, let $\mathcal{E}_R^{s,a}$ be not strictly minimal. By definition, there is a distribution ρ over the remaining actions in s such that $\mathcal{E}_R^{s,\rho} \subseteq \mathcal{E}_R^{s,a}$. Then, this distribution ρ must satisfy $B\rho + d \geq 0$. \square

Complexity Again let $|S| = n$, $|A| = m$, b be the maximal number of different actions $\max_{s \in S} |\{I(s, a, \cdot) \mid a \in A\}|$, and f be the maximal support of an action $\max_{s \in S, a \in A} |\{s' \mid I(s, a, s') \neq [0, 0]\}|$.

Again, `Violate(\exists \forall)` is called at most n^3 times. The procedure `Violate(\exists \forall)` is then linear in m and in the complexity of `Minimal`. There are at most $(f \cdot 2^{f-1})^b$ combinations of corners of the polytopes. For each such combination, $b(b-1)$ times the intersection points of a line and a polytope are computed (in time polynomial in $|\mathfrak{M}|$), and at most $f!$ facets of the resulting polytope R are inspected. Overall, computing of $\sim_{(\exists \forall)}$ can be done in time $|\mathfrak{M}|^{\mathcal{O}(1)} \cdot 2^{\mathcal{O}(f^2 b)}$.

5 Case Study

As a case study, we consider a model of Carrier Sense Multiple Access with Collision Detection (CSMA / CD), which is an access control on a shared medium, used mostly in early Ethernet technology. In this scheme multiple devices can be connected to a shared bus. Multiple attempts at the same time to grab bus access leads to collision. At this point, the senders in collision probabilistically schedule a retransmission according to exponential back-off algorithm. The algorithm uniformly determines a delay before the next retransmission, which is between 0 to $2^n - 1$ time slots after occurrence of n -th collision. After a pre-specified number of failed retransmissions, a sender aborts the sending process.

There are two sources of uncertainty in the model. Uncertainty in sending data lies in the fact that the exact probability of sending a message from a sender could be unknown. Instead it is within an interval. The other source comes from imprecise information about collision. If two nodes try to send a frame at the slightly same time, a collision will happen. Conversely it will not happen, when the later transmitter checks the bus and detects it occupied. Since the exact probability of a collision occurrence depends on many parameters and is likely unknown, it is expressed as an uncertain interval rather than an exact value in the model.

Concurrent execution of the node and the bus processes assembles the CSMA/CD model. To this end we need a formalism that supports communication among components via parallel composition. We thus consider a subclass of abstract PAs [9] with interval constraints on probabilities. The subclass in general is not closed under parallel composition. The problem arises when two actions exhibiting uncertainty want to synchronise. Parallel composition in this case imposes some interdependency between the choices of the composed action, which cannot be expressed by a simple interval bound and needs to be expressed by more complicated polynomial constraints. Nevertheless by excluding synchronisation of actions containing uncertainty, abstract PAs with interval constraints feature closure under parallel composition and thereby allow compositional modelling. This is of course not a strict restriction, because we can always shift uncertainty to the actions that are not subject to parallel composition by introducing proper auxiliary states and transitions. In our case study all components are in this subclass and respects the restriction, as uncertainty prevails on actions that are not subject to parallel composition. Consequently it enables us to utilise compositional system design by using existing tools. Since the model arising from parallel composition is not subject to any further communication, we can close it and obtain an IMDP at the end.

We use process algebra prCRL [21], implemented in tool scoop [35], for compositional modelling of CSMA/CD. The model has two parameters: number of nodes attached to the bus and maximum collision allowed before abortion. As we are interested in model checking of a model arisen from parallel composition we apply the semantics of bisimulation in cooperative way, namely $\sim_{(\vee)}$. The state space is generated by scoop and then the bisimulation quotient is computed. Since the maximum size f of the set supported by uncertain transitions is two, the algorithm of Section 4 is tractable. Reduction in state and transition space gained after bisimulation minimisation is reported in Table 1. As shown in the table, the reduction of both state and transition space increases when putting more nodes in the network. Indeed, then there are more nodes performing similar activities and thereby increasing the symmetry in the model. On the other hand, increasing the maximum number of collisions allows the nodes to more likely send frames at different time slots. As a result it decreases the symmetry and then the reduction factor.

Table 1: Impact of bisimulation minimisation on CSMA/CD model²

Node #	Max collision #	Original Model		Minimised Model		Reduction Factor	
		State #	Transition #	States #	Transition #	For states	For transitions
2	1	233	466	120	220	48%	53%
	2	497	988	310	581	38%	41%
	3	865	1858	576	1186	33%	36%
3	1	4337	10074	1244	2719	71%	73%
	2	52528	125715	18650	42795	64%	66%
	3	239213	619152	90492	225709	62%	64%
4	1	60357	154088	10904	27308	82%	82%
	2	1832005	4876636	421570	1112129	77%	77%
5	1	751905	2043090	90538	248119	88%	88%

6 Conclusion

In this paper, we study strong bisimulations for interval MDPs. In these models there are two sources of non-determinism and we deal with different interpretations of these non-determinisms. This yields two different bisimulations and we give decision algorithms for both of them.

Note that our decision algorithms can be easily adapted to the slightly broader setting of uncertain MDPs with rectangular uncertainty sets [25]. In this setting, a general convex polytope (not necessarily induced by intervals) is associated to each action in each state. Still, it is assumed that transition probabilities from different states or under different actions are independent.

First open question for future work is the exact complexity of our decision problems. One way to address this question is to prove NP-hardness of the general problem. Another way is to identify interesting subclasses of interval MDPs for that a polynomial-time algorithm exists. Second direction for future work is to address a richer formalism for uncertainties (such as polynomial constraints or even parameters appearing in multiple states/actions). Third, compositional modelling over interval models also deserves a more systematic treatment. Understanding better the ways how large models with interval uncertainties can be composed, may bring further ideas for efficient analysis of these models.

Acknowledgements We would like to thank Holger Hermanns for inspiring discussions. This work has been supported by the DFG as part of the SFB/TR 14 “Automatic Verification and Analysis of Complex Systems” (AVACS), by the Czech Science Foundation under the grant agreement no. P202/12/G061, by the DFG/NWO Bilateral Research Programme ROCKS, and by the European Union Seventh Framework Programme under grant agreement no. 295261 (MEALS) and 318490 (SENSATION).

References

- [1] Rajeev Alur, Thomas A. Henzinger, Orna Kupferman & Moshe Y. Vardi (1998): *Alternating Refinement Relations*. In: *CONCUR, LNCS 1466*, Springer, pp. 163–178. Available at <http://dx.doi.org/10.1007/BFb0055622>.
- [2] Michael Benedikt, Rastislav Lenhardt & James Worrell (2013): *LTL Model Checking of Interval Markov Chains*. In: *TACAS, LNCS 7795*, Springer, pp. 32–46. Available at http://dx.doi.org/10.1007/978-3-642-36742-7_3.

² The computation time does not reflect the complexity of the algorithm, as it is greatly effected by the file exchange between the tools used for modelling and bisimulation minimisation. Hence it is omitted from the table.

- [21] Joost-Pieter Katoen, Jaco van de Pol, Mariëlle Stoelinga & Mark Timmer (2010): *A Linear Process-Algebraic Format for Probabilistic Systems with Data*. In: *ACSD*, IEEE Computer Society, pp. 213–222. Available at <http://doi.ieeecomputersociety.org/10.1109/ACSD.2010.18>.
- [22] Igor Kozine & Lev V. Utkin (2002): *Interval-Valued Finite Markov Chains*. *Reliable Computing* 8(2), pp. 97–113. Available at <http://dx.doi.org/10.1023/A:1014745904458>.
- [23] Kim Guldstrand Larsen & Arne Skou (1991): *Bisimulation through Probabilistic Testing*. *Inf. Comput.* 94(1), pp. 1–28. Available at [http://dx.doi.org/10.1016/0890-5401\(91\)90030-6](http://dx.doi.org/10.1016/0890-5401(91)90030-6).
- [24] Robin Milner (1989): *Communication and concurrency*. PHI Series in computer science, Prentice Hall.
- [25] Arnab Nilim & Laurent El Ghaoui (2005): *Robust Control of Markov Decision Processes with Uncertain Transition Matrices*. *Operations Research* 53(5), pp. 780–798. Available at <http://dx.doi.org/10.1287/opre.1050.0216>.
- [26] Robert Paige & Robert E. Tarjan (1987): *Three Partition Refinement Algorithms*. *SIAM J. on Computing* 16(6), pp. 973–989. Available at <http://dx.doi.org/10.1137/0216062>.
- [27] Amir Pnueli (1977): *The Temporal Logic of Programs*. In: *FOCS*, IEEE Computer Society, pp. 46–57. Available at <http://doi.ieeecomputersociety.org/10.1109/SFCS.1977.32>.
- [28] Alberto Puggelli, Wenchao Li, Alberto L. Sangiovanni-Vincentelli & Sanjit A. Seshia (2013): *Polynomial-Time Verification of PCTL Properties of MDPs with Convex Uncertainties*. In: *CAV, LNCS 8044*, Springer, pp. 527–542. Available at http://dx.doi.org/10.1007/978-3-642-39799-8_35.
- [29] Martin L. Puterman (1994): *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, 1st edition. John Wiley & Sons, Inc., New York, NY, USA. Available at <http://dx.doi.org/10.1002/9780470316887>.
- [30] Matthias Ringwald (2009): *Reducing Uncertainty in Wireless Sensor Networks - Network Inspection and Collision-Free Medium Access*. Ph.D. thesis, ETH Zurich, Zurich, Switzerland.
- [31] Roberto Segala (1995): *Modeling and Verification of Randomized Distributed Real-Time Systems*. Ph.D. thesis, Laboratory for Computer Science, Massachusetts Institute of Technology.
- [32] Roberto Segala & Nancy A. Lynch (1994): *Probabilistic Simulations for Probabilistic Processes*. In: *CONCUR, LNCS 836*, pp. 481–496. Available at <http://dx.doi.org/10.1007/BFb0015027>.
- [33] Koushik Sen, Mahesh Viswanathan & Gul Agha (2006): *Model-Checking Markov Chains in the Presence of Uncertainties*. In: *TACAS, LNCS 3920*, Springer, pp. 394–410. Available at http://dx.doi.org/10.1007/11691372_26.
- [34] K. Subramani (2009): *On the Complexities of Selected Satisfiability and Equivalence Queries over Boolean Formulas and Inclusion Queries over Hulls*. *JAMDS 2009*. Available at <http://dx.doi.org/10.1155/2009/845804>.
- [35] Mark Timmer (2011): *SCOOP: A Tool for Symbolic Optimisations of Probabilistic Processes*. In: *QEST*, IEEE Computer Society, pp. 149–150. Available at <http://doi.ieeecomputersociety.org/10.1109/QEST.2011.27>.
- [36] Eric M. Wolff, Ufuk Topcu & Richard M. Murray (2012): *Robust control of uncertain Markov Decision Processes with temporal logic specifications*. In: *CDC, IEEE*, pp. 3372–3379. Available at <http://dx.doi.org/10.1109/CDC.2012.6426174>.
- [37] Di Wu & Xenofon D. Koutsoukos (2008): *Reachability analysis of uncertain systems using bounded-parameter Markov decision processes*. *Artif. Intell.* 172(8-9), pp. 945–954. Available at <http://dx.doi.org/10.1016/j.artint.2007.12.002>.
- [38] Wang Yi (1994): *Algebraic Reasoning for Real-Time Probabilistic Processes with Uncertain Information*. In: *FTRTFT, LNCS 863*, Springer, pp. 680–693. Available at http://dx.doi.org/10.1007/3-540-58468-4_190.