# POWVER

# Technical Report 2016-02

Title: **Exploiting Robust Optimization for Interval Probabilistic Bisimulation**

Author: Ernst Moritz Hahn, Vahid Hashemi, Holger Hermanns, Andrea Turrini

Report Number: 2016-02

ERC Project: Power to the People. Verified.

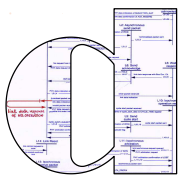
ERC Project ID: 695614

Funded Under: H2020-EU.1.1. – EXCELLENT SCIENCE

Host Institution: Universität des Saarlandes, Dependable Systems and Software

Published In: QEST 2016





POWER TO THE PEOPLE.
VERIFIED.
erc

# Exploiting Robust Optimization for Interval Probabilistic Bisimulation

Ernst Moritz Hahn[1], Vahid Hashemi[2,3], Holger Hermanns[3], and Andrea Turrini[1]

[1] State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing, China
[2] Max Planck Institute for Informatics, Saarbrücken, Germany
[3] Department of Computer Science, Saarland University, Saarbrücken, Germany

**Abstract.** Verification of PCTL properties of MDPs with convex uncertainties has been investigated recently by Puggelli et al. However, model checking algorithms typically suffer from the state space explosion problem. In this paper, we discuss the use of probabilistic bisimulation to reduce the size of such an MDP while preserving the PCTL properties it satisfies. As a core part, we show that deciding bisimilarity of a pair of states can be encoded as adjustable robust counterpart of an uncertain LP. We show that using *affine decision rules*, probabilistic bisimulation relation can be approximated in polynomial time. We have implemented our approach and demonstrate its effectiveness on several case studies.

## 1 Introduction

Real world systems are usually too complex to be analyzed in full detail. To reduce the complexity of such an analysis, a simplified but accurate enough model of the system has to be constructed and then verified with respect to a number of properties the system is expected to satisfy. Among others, probability, nondeterminism, and uncertainty are core aspects of a real world system that are worth considering in the model. *Probability* represents the fact that the behaviour of the system is not uniquely determined by its status and the action it performs, but depends on random choices as well; these choices may be present by design (as the toss of a coin in a distributed algorithm so as to break symmetry) or to represent general properties such as transmission errors during a communication. *Nondeterminism* can be used whenever a specific behavior is unknown or it is left undetermined by purpose: an example of the former is the unknown relative speed of several distributed systems interacting with each other while an example of the latter is the possibility of leaving some behavior undetermined so an implementation can fix it. *Uncertainty* appears when some information is available but it is not precise enough to be represented as a probability.

A problem that may occur during the formal verification of a system, for instance by model checking it, is the notorious *state-explosion* problem. Such a problem can be mitigated by reducing the size of the model to be verified while preserving its properties. This goal can by achieved by finding another model

that is smaller than the original one while behaving the same. Bisimulation allows us to construct such a model; this strategy has been proven very effective [16,29] in related settings.

Several models have been proposed in literature as frameworks for modelling real world systems, frameworks equipped with bisimulation. Among others, there are Labelled Transition Systems, Probabilistic Automata [42], and Markov Decision Processes (*MDP*s). In this work we focus on the Interval Markov Decision Processes (*IMDP*s) model [27,28,38,41,45,46], an extension of classical *MDP*s where uncertainty is represented by intervals of probability values. It is known that that bisimilar *IMDP*s satisfy the same PCTL properties [27]. As established in [27,28], computing the coarsest bisimulation on a given *IMDP* is a difficult problem; our aim is to provide a polynomial algorithm that returns a non-trivial bisimulation for the given *IMDP*. We achieve this goal by taking advantage of the results from the Operations Research community about robust optimization and uncertain Linear Programming (LP) problems.

Summarizing, the main contributions of this paper are as follows.

- We build a bridge between *Probabilistic Verification* and *Robust Optimization* and establish a novel modelling of the probabilistic bisimulation problem for interval *MDP*s as an instance of an uncertain LP problem.
- We show that, by using *affine decision rules*, the probabilistic bisimulation problem for *IMDP*s can be approximately decided in polynomial time.
- We show promising results on a number of case studies, obtained by a prototypical implementation of our algorithm.

*Related work.* We classify related works in four areas. Firstly, various probabilistic modelling formalisms with uncertain transitions are studied in the literature. Interval Markov chains [31,35] or Abstract Markov chains [20] extend standard discrete-time Markov chains (MCs) with interval uncertainties and thus do not feature the non-deterministic choice of transitions. Uncertain MDPs [38,40,45] allow for more general sets of distributions to be associated with each transition, not only those described by intervals. Usually, this is restricted to *rectangular uncertainty sets* requiring that the uncertainty is linear and independent for any two transitions of any two states. Our general algorithm working with polytopes can be easily adapted to this setting. Parametric MDPs [26] to the contrary allow for such dependencies as every probability is described as a rational function of a finite set of global parameters.

Secondly, computational complexity of the probabilistic bisimulation for uncertain probabilistic models has been studied quite recently in [27,28]. Among similar concepts studied in the literature are simulation [22, 47] and refinement [18,19,31] relations for previously mentioned models.

Thirdly, from model checking viewpoint, many new verification algorithms for interval models appeared in last few years. Reachability and expected total reward is addressed for Interval MCs [15] as well as Interval MDPs [46]. PCTL model checking and LTL model checking are studied for Interval MCs [9,14,15] and also for Interval MDPs [41,45]. Among other technical tools, all these approaches make use of (robust) dynamic programming relying on the fact that

transition probability distributions are resolved dynamically: a probability distribution is chosen from interval restrictions each time the system enters a state. For the static resolution of distributions, an adaptive discretization technique for PCTL parameter synthesis is given in [26]. Uncertain models are also widely studied in the control community [23, 38, 46], mainly interested in maximal expected finite-horizon reward or maximal expected discounted reward.

Finally, as regards the application of *Robust Optimization* in *Probabilistic Verification* community, to the best of our knowledge, we are not aware of any work in the literature. Therefore, the current contribution is novel in this matter. On the other hand, the aforementioned theory has been adapted and applied successfully in control theory realm. For instance, Abate et al. [5] developed a robust modal predictive control using two-stage robust optimization.

## 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. We denote by $\mathbb{I}$ the set of closed sub-intervals of $[0,1]$ and, for a given $[a,b] \in \mathbb{I}$, we denote by $\inf[a,b]$ the lower bound $a$ and by $\sup[a,b]$ the upper bound $b$. We denote by $b^k$ the $k$-th element of a vector $b \in \mathbb{R}^n$. For a set $X$, we denote by $\Delta(X)$ the set of discrete probability distributions over $X$; given $\rho \in \Delta(X)$, we denote by $\mathrm{Supp}(\rho) = \{\, x \in X \mid \rho(x) > 0 \,\}$ the support of $\rho$ and we say that $\rho$ is Dirac, denoted $\delta_x$, if $\mathrm{Supp}(\rho) = \{x\}$ with $x \in X$. For an equivalence relation $\mathcal{R}$ on $X$ and $\rho_1, \rho_2 \in \Delta(X)$, we write $\rho_1 \; \mathcal{L}(\mathcal{R}) \; \rho_2$ if for each $\mathcal{C} \in X/\mathcal{R}$, it holds that $\rho_1(\mathcal{C}) = \rho_2(\mathcal{C})$. By abuse of notation, we extend $\mathcal{L}(\mathcal{R})$ to distributions over $X/\mathcal{R}$, i.e., for $\rho_1, \rho_2 \in \Delta(X/\mathcal{R})$, we write $\rho_1 \; \mathcal{L}(\mathcal{R}) \; \rho_2$ if for each $\mathcal{C} \in X/\mathcal{R}$, it holds that $\rho_1(\mathcal{C}) = \rho_2(\mathcal{C})$.

### 2.1 Interval Markov Decision Processes

Let us formally define Interval Markov Decision Processes.

**Definition 1 (*IMDP*s).** *An* Interval Markov Decision Process *(IMDP)* $\mathcal{M}$ *is a tuple* $\mathcal{M} = (S, \bar{s}, \mathcal{A}, \mathtt{AP}, L, I)$*, where* $S$ *is a finite set of* states*,* $\bar{s} \in S$ *is the* initial state*,* $\mathcal{A}$ *is a finite set of* actions*,* $\mathtt{AP}$ *is a finite set of* atomic propositions*,* $L \colon S \to 2^{\mathtt{AP}}$ *is a* labeling function*, and* $I \colon S \times \mathcal{A} \times S \to \mathbb{I}$ *is an* interval transition probability function*.*

Given $s \in S$ and $a \in \mathcal{A}$, we write $s \xrightarrow{a} \mu_s$ whenever $\mu_s \in \Delta(S)$ is a *feasible distribution*, i.e., for each $s' \in S$ we have $\mu_s(s') \in I(s,a,s')$. Let $\mathcal{P}^{s,a} = \{\, \mu_s \in \Delta(S) \mid s \xrightarrow{a} \mu_s \,\}$; we denote by $\mathcal{A}(s) = \{\, a \in \mathcal{A} \mid \mathcal{P}^{s,a} \neq \emptyset \,\}$ the set of actions that are enabled from $s$ and we require that $\mathcal{A}(s) \neq \emptyset$ for each $s \in S$.

We extend $I$ to sets of states as follows: given $S' \subseteq S$, we let

$$I(s,a,S') = \left[ \min\left\{1, \sum_{s' \in S'} \inf I(s,a,s')\right\}, \min\left\{1, \sum_{s' \in S'} \sup I(s,a,s')\right\} \right].$$

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

Let us define the semantics of an *IMDP* formally. A *path* is a finite or infinite sequence of states $\pi = s_1 s_2 \ldots$. For a finite path $\pi$, we denote by $last(\pi)$ the last state of $\pi$. The set of all finite and infinite paths are denoted by $Paths^*$ and $Paths^\omega$, respectively. Furthermore, let $Cyl_\pi = \{\, \pi' \in Paths^\omega \mid \pi \leqslant \pi' \,\}$ denote the set of paths having $\pi \in Paths^*$ as prefix.

**Definition 2 (Scheduler and Nature).** *Given an IMDP $\mathcal{M}$, a* scheduler *is a function $\sigma\colon Paths^* \to \Delta(\mathcal{A})$ such that for each $\pi \in Paths^*$, $\mathrm{Supp}(\sigma(\pi)) \subseteq \mathcal{A}(last(\pi))$. Further, a* nature *is a function $\nu\colon Paths^* \times \mathcal{A} \to \Delta(S)$ such that for each $\pi \in Paths^*$ and $a \in \mathcal{A}(last(\pi))$, $\nu(\pi, a) \in \mathcal{P}^{last(\pi),a}$. We denote by $\mathfrak{S}$ and $\mathfrak{N}$ the set of all schedulers and natures of $\mathcal{M}$, respectively.*

For an initial state $s$, a scheduler $\sigma$, and a nature $\nu$, let $Pr_s^{\sigma,\nu}$ denote the unique probability measure over $(Paths^\omega, \mathcal{B})$[4] such that the probability $Pr_s^{\sigma,\nu}[Cyl_{s'}]$ of starting in $s'$ equals 1 if $s' = s$ and 0 otherwise and the probability $Pr_s^{\sigma,\nu}[Cyl_{\pi s'}]$ of traversing a finite path $\pi s'$ equals $Pr_s^{\sigma,\nu}[Cyl_\pi] \cdot \sum_{a \in \mathcal{A}} \sigma(\pi)(a) \cdot \nu(\pi, a)(s')$.

Observe that the scheduler does not choose an action but a *distribution* over actions. It is well-known [42] that such a randomization is useful in the context of bisimulations as it allows to define coarser equivalence relations. To the contrary, nature is not allowed to randomize over the set of feasible distributions $\mathcal{P}^{s,a}$. This is in fact not necessary, since the set $\mathcal{P}^{s,a}$ is closed under convex combinations. Finally, we call a scheduler $\sigma$ *deterministic*, or *Dirac* if, for each finite path $\pi \in Paths^*$, $\sigma(\pi)$ is a Dirac distribution.

We determine the size of an *IMDP* $\mathcal{M}$ as follows. Let $|S|$ denote the number of states in $\mathcal{M}$; each state has at most $|\mathcal{A}|$ actions and at most $|\mathcal{A}| \cdot |S|$ transitions, each of which is associated with a probability interval. Therefore, the overall size of $\mathcal{M}$ is $|\mathcal{M}| \in \mathcal{O}(|S|^2 \cdot |\mathcal{A}|)$.

### 2.2 Robust Optimization

Robust optimization is a new approach in mathematical optimization that is concerned about optimization problems in which a certain level of robustness is desirable against *uncertainty* [6,7]. This modelling methodology is integrated with computational tools to treat optimization problems with uncertain data that is only known to be included in some uncertainty set [3, 24, 37]. This approach has been shown to be very useful in real-world applications that are entirely or to a certain extent affected by uncertainty [8,10]. In this section, we

[4] Here, $\mathcal{B}$ is the standard $\sigma$-algebra over $Paths^\omega$ generated from the set of all cylinder sets $\{Cyl_\pi \mid \pi \in Paths^*\}$. The unique probability measure is obtained by the application of the extension theorem (see, e.g., [11]).

introduce the concept of Uncertain Linear Programming problems (ULPs) and afterwards, we provide an overview of the essential background required for the rest of the paper. We refer the interested reader to [6, 10] for a comprehensive reference on robust optimization.

**Uncertain Linear Programming (ULPs)** Linear Programming (LP) problems are problems that can be described in canonical form as:

$$\mathrm{Min}_{x\in\mathbb{R}^n}\left\{c^T x : Ax \leq b\right\}$$

where $x \in \mathbb{R}^n$ is the vector of *decision variables*, $c \in \mathbb{R}^n$ is the vector of *coefficients*, $A \in \mathbb{R}^{m\times n}$ is the constant *coefficient matrix* and $b \in \mathbb{R}^m$ is the *right hand side* vector.

The data of an LP problem, i.e., the collection of tuples $[c, A, b]$, in general are not known precisely when the LP encodes a real-world problem. This issue reveals the need for an approach to produce LP solutions which are immune against uncertainty.

**Definition 3 (cf. [6, 7]).** *An Uncertain Linear Program (ULP) is a family*

$$\left\{\mathrm{Min}_{x\in\mathbb{R}^n}\{c^T x : Ax \leq b\}\right\}_{[c,A,b]\in\mathcal{Z}} \tag{1}$$

*of LP problems* $\mathrm{Min}_{x\in\mathbb{R}^n}\{c^T x : Ax \leq b\}$ *with the same structure (i.e., same number of constraints and variables) in which the data range over a given nonempty compact uncertainty set* $\mathcal{Z} \subset \mathbb{R}^n \times \mathbb{R}^{m\times n} \times \mathbb{R}^m$.

To simplify the notation, we may write $\left\{\mathrm{Min}\{c^T x : Ax \leq b\}\right\}_{\mathcal{Z}}$.

In contrast to an usual single LP problem, it is not possible to associate the notions of feasibility/optimal solutions and optimal objective value with a collection of optimization problems like ULPs. In the setting of ULPs, the feasible solutions are solutions which are *robust feasible*. Roughly speaking, feasible solutions are those which satisfy the set of constraints whatever the realization of uncertain data is. More precisely:

**Definition 4 (cf. [6,8]).** *A vector* $x \in \mathbb{R}^n$ *is* robust feasible *to an ULP with uncertainty set* $\mathcal{Z}$ *if for each* $[c, A, b] \in \mathcal{Z}$, $Ax \leq b$. *Given a robust feasible solution* $x$, *the* robust value $\hat{z}(x)$ *of the objective function is* $\hat{z}(x) := \sup_{[c,A,b]\in\mathcal{Z}} c^T x$.

After carefully defining the robust feasible/optimal solutions as well as their robust objective value, we can describe the central concept in robust optimization setting that is the *robust counterpart* (RC) of an uncertain LP problem.

**Definition 5 (cf. [8]).** *Given an ULP problem* $\left\{\mathrm{Min}\{c^T x : Ax \leq b\}\right\}_{\mathcal{Z}}$, *the* Robust Counterpart *of ULP is the optimization problem*

$$\mathrm{Min}_{x\in\mathbb{R}^n}\left\{\hat{z}(x) = \sup_{[c,A,b]\in\mathcal{Z}}\{c^T x : Ax \leq b\}\right\}$$

*that seeks for the best possible value of the objective function among all possible robust feasible solutions to the ULP. Furthermore, the optimal solution/value to the robust counterpart is called the robust optimal solution/value to the ULP.*

In the robust counterpart (RC) approach, all the variables are "*here and now decisions*": they must be decided before the realization of unknown data. However, in some cases, some part of the variables are "*wait and see decisions*", i.e., they might tune themselves to the varying parameters. In the rest of the paper, we call the variables that may depend on the realizations of the uncertain data as *adjustable*, while other variables are called *non-adjustable*. Therefore, we can split the vector $x$ of Eq. (1) from Def. 3 as $x = (u, v)^T$ where the sub-vectors $u$ and $v$ indicate the non-adjustable and the adjustable variables, respectively.

**Adjustable Robust Counterpart** Splitting the decision variable $x$ to the adjustable and non-adjustable variables allows us to rewrite the uncertain LP (1) as the following equivalent form:

$$\left\{\mathrm{Min}_{u,v}\{c^T u : Uu + Vv \leq b\}\right\}_{[c,U,V,b]\in\mathcal{Z}} \tag{2}$$

In the above presentation, without loss of generality, we assume that the objective function is normalized with respect to the non-adjustable variables. Moreover, the matrix $V$ is called *recourse matrix* [17] and when it is not uncertain, we call the uncertain LP (2) a *fixed recourse* one. We can now define the RC and the *Adjustable robust counterpart (ARC)* as follows:

$$\text{RC: } \mathrm{Min}_u\{c^T u : \exists v : \forall [U,V,b] \in \mathcal{Z} : Uu + Vv \leq b\}; \tag{3}$$

$$\text{ARC: } \mathrm{Min}_u\{c^T u : \forall [U,V,b] \in \mathcal{Z} : \exists v : Uu + Vv \leq b\}. \tag{4}$$

It is not difficult to see that ARC is less conservative than RC allowing for better optimal values while still having all realizations of the constraints satisfied. The distinction between RC and ARC can be very significant (see, e.g., [6, 7]).

The RC of an uncertain LP is a computationally tractable problem in general [8]. On the contrary, this is not the case with ARC. This fact stimulates a very good reason to introduce the notion of *Affinely Adjustable Robust Counterpart (AARC)* of an uncertain LP in which we make a simplification on how the adjustable variables can tune themselves upon the uncertain data. By posing $v = w + W\xi$, we consider an affine dependency between adjustable variables and uncertain parameter. Therefore, the AARC of the uncertain LP (2) reads as:

$$\begin{aligned} &\mathrm{Min}_{u,w,W}\left\{c^T u : Uu + V(w + W\xi) \leq b, \forall(\xi \equiv [U,V,b] \in \mathcal{Z})\right\} \\ \equiv\ &\mathrm{Min}_u\left\{c^T u : \forall(\xi \equiv [U,V,b] \in \mathcal{Z}) : \exists(w,W) : Uu + V(w + W\xi) \leq b\right\}. \end{aligned} \tag{5}$$

## 3 Probabilistic Bisimulation for Interval MDPs

This section revisits required main results on probabilistic bisimulation for interval MDPs, as developed in [27]. In the setting of this paper, we consider the notion of probabilistic bisimulation for the cooperative interpretation of interval MDPs. This semantics is very natural in the context of verification of parallel

systems with uncertain transition probabilities in which we assume that scheduler and nature are resolved *cooperatively* in the most *adversarial* way: in the game view of the bisimulation, challenging scheduler and nature work together in order to defeat the defender with a transition that can not be matched.

Besides the cooperative behaviour, the choice of a probability distribution respecting the interval constraints can be done either *statically* [31], i.e., at the beginning once for all, or *dynamically* [30,43], i.e., independently at each computation step. In this paper, we focus on the dynamic approach in resolving the stochastic nondeterminism: it is easier to work with algorithmically and can be seen as a relaxation of the static approach that is often intractable [9,14,19,23].

Let $s \longrightarrow \mu_s$ denote a transition from $s$ to $\mu_s$ taken cooperatively, i.e., there is a scheduler $\sigma \in \mathfrak{S}$ and a nature $\nu \in \mathfrak{N}$ such that $\mu_s = \sum_{a \in \mathcal{A}} \sigma(s)(a) \cdot \nu(s,a)$. In other words, $s \longrightarrow \mu_s$ if $\mu_s \in \mathrm{CH}(\bigcup_{a \in \mathcal{A}(s)} \mathcal{P}^{s,a})$ where $\mathrm{CH}(X)$ denotes the convex hull of $X$.

**Definition 6 (cf. [27]).** *Given an IMDP $\mathcal{M}$, let $\mathcal{R} \subseteq S \times S$ be an equivalence relation. We say that $\mathcal{R}$ is a* probabilistic bisimulation *if for each $(s,t) \in \mathcal{R}$ we have that $L(s) = L(t)$ and for each $s \longrightarrow \mu_s$ there exists $t \longrightarrow \mu_t$ such that $\mu_s \, \mathcal{L}(\mathcal{R}) \, \mu_t$. Furthermore, we write $s \sim_c t$ if there is a probabilistic bisimulation $\mathcal{R}$ such that $(s,t) \in \mathcal{R}$.*

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 $t$. It is shown in [27] that the bisimulation $\sim_c$ preserves the (cooperative) universally quantified PCTL satisfaction $\models_c$.

**Theorem 7 (cf. [27]).** *For states $s \sim_c t$ and any PCTL formula $\varphi$, we have $s \models_c \varphi$ if and only if $t \models_c \varphi$.*

Computation of probabilistic bisimulation for *IMDP*s follows the standard partition refinement approach [13,32,39]. However, the core part of the algorithm is to find out whether two states "violate the definition of bisimulation". Verification of this violation amounts to checking inclusion of polytopes defined as follows. For $s \in S$ and $a \in \mathcal{A}(s)$, recall that $\mathcal{P}^{s,a}$ denotes the polytope of feasible successor distributions over *states* with respect to taking the action $a$ in the state $s$. By $\mathcal{P}^{s,a}_{\mathcal{R}}$, we denote the polytope of feasible successor distributions over *equivalence classes* of $\mathcal{R}$ with respect to taking the action $a$ in the state $s$. Formally, for $\mu \in \Delta(S/\mathcal{R})$ we set $\mu \in \mathcal{P}^{s,a}_{\mathcal{R}}$ if, for each $\mathcal{C} \in S/\mathcal{R}$, we have $\mu(\mathcal{C}) \in I(s,a,\mathcal{C})$. Furthermore, we define $\mathcal{P}^{s}_{\mathcal{R}} = \mathrm{CH}(\bigcup_{a \in \mathcal{A}(s)} \mathcal{P}^{s,a}_{\mathcal{R}})$. It is the set of feasible successor distributions over $S/\mathcal{R}$ with respect to taking an *arbitrary* distribution over actions in state $s$. As specified in [27], checking violation of a given pair of states, amounts to check equality of the corresponding constructed polytopes for the states. As regards the computational complexity of the proposed algorithm, the following theorem indicates that it is fixed parameter tractable.

**Theorem 8 (cf. [27]).** *Given an IMDP $\mathcal{M}$, let $f$ be the maximal fanout, i.e., $f = \max_{s \in S, a \in \mathcal{A}(s)} |\{\, s' \in S \mid I(s,a,s') \neq [0,0] \,\}|$. Computing $\sim_c$ can be done in time $|\mathcal{M}|^{\mathcal{O}(1)} \cdot 2^{\mathcal{O}(f)}$.*

The exact time complexity of deciding probabilistic bisimulation for *IMDP*s has recently been explored in [28], leading to the following result.

**Theorem 9.** *Given an IMDP $\mathcal{M}$, computing $\sim_c$ is coNP-complete.*

## 4 Computational Tractability

Def. 6 is the central definition around which the paper revolves. Given an *IMDP*, the complexity of computing $\sim_c$ strictly depends on finding $t \longrightarrow \mu_t$: we show how a finer (sub-optimal) equivalence relation can be computed in polynomial time. The bisimulation in Def. 6 can be reformulated equivalently as follows:

**Definition 10.** *Let $\mathcal{R} \subseteq S \times S$ be an equivalence relation. We say that $\mathcal{R}$ is a* probabilistic bisimulation *if $(s,t) \in \mathcal{R}$ implies that $L(s) = L(t)$ and for each $a \in \mathcal{A}(s)$ and each $\mu_s \in \mathcal{P}_{\mathcal{R}}^{s,a}$, there exists $\mu_t \in \mathcal{P}_{\mathcal{R}}^{t}$ such that $\mu_s \, \mathcal{L}(\mathcal{R}) \, \mu_t$.*

Recall that a probabilistic bisimulation can be seen as a game between two players: in each round, the challenger, or attacker, $s$ proposes a transition, or step, that has to be matched by the defender $t$. The two states $s$ and $t$ are bisimilar if the defender is always able to match the challenging transitions proposed by the attacker, that is, the game can be played forever. Correspondingly, in our setting, probabilistic bisimulations require that each transition proposed by the challenger $s$ which is selected from the set $\mathcal{P}_{\mathcal{R}}^{s,a}$, is matched by the defender $t$ via a single (combined) transition. The above definition essentially disallows the state $s$ to randomize over the set of its available actions. Therefore, instead of allowing the challenger to pick a probability distribution from $\mathrm{CH}(\bigcup_{a \in \mathcal{A}(s)} \mathcal{P}_{\mathcal{R}}^{s,a})$, we restrict his choice to select a distribution for an action from the polytope $\mathcal{P}_{\mathcal{R}}^{s,a}$. This restriction does not lead to any loss of generality, since it is routine to check that the bisimulation $\mathcal{R}$ from Def. 10 satisfies the condition of Def. 6.

### 4.1 Robust Methodologies for Probabilistic Bisimulation

We now discuss the key elements of a decision algorithm for probabilistic bisimulation on *IMDP*s. As we will see in Sec. 5, the core part—and the main source of the exponential complexity of the decision algorithm in [27]—is the need to repeatedly verify the step condition, that is, given a challenging transition $\mu \in \mathcal{P}_{\mathcal{R}}^{s}$ and $(s,t) \in \mathcal{R}$, to check if there exists $t \longrightarrow \mu_t$ such that $\mu \, \mathcal{L}(\mathcal{R}) \, \mu_t$. We show that, using some inspiration from network flow problems, it is possible to treat a transition $t \longrightarrow \mu_t$ of the *IMDP* $\mathcal{M}$ as a flow where the initial probability mass $\delta_t$ flows and splits along transitions appropriately to the transition target distributions and the resolution of the nondeterminism fulfilled by the scheduler and nature. This intuition essentially enables us to model the probabilistic bisimulation problem as an *adjustable robust counterpart* of an uncertain LP problem that is intractable in general [6, 7].

### 4.2 Adjustable Robust Counterpart for Probabilistic Bisimulation

From now on, we assume that the *IMDP* $\mathcal{M}$, the state $t$, the probability distribution $\mu$, and the equivalence relation $\mathcal{R}$ on $S$ are given. We intend to verify or refute the existence of a transition $t \longrightarrow \mu_t$ of $\mathcal{M}$ satisfying $\mu\ \mathcal{L}(\mathcal{R})\ \mu_t$ via the construction of a flow through the network graph $G(t,\mathcal{R}) = (V,E)$ defined as follows: the set of vertices is $V = \{\triangle, \blacktriangledown, t\} \cup S_{\mathcal{A}} \cup S_{\mathcal{R}} \cup (S/\mathcal{R})$ where $S_{\mathcal{A}} = \{\, t_a \mid a \in \mathcal{A}(t) \,\}$ and $S_{\mathcal{R}} = \{\, s_{\mathcal{R}} \mid s \in S \,\}$, and the set of arcs is $E = \{(\triangle, t)\} \cup \{\, (v_{\mathcal{R}}, \mathcal{C}), (\mathcal{C}, \blacktriangledown) \mid \mathcal{C} \in S/\mathcal{R}, v \in \mathcal{C} \,\} \cup \{\, (t, t_a), (t_a, v_{\mathcal{R}}) \mid a \in \mathcal{A}(t), v \in S \,\}$. In the flow network definition, $\triangle$ and $\blacktriangledown$ are the source node and the sink node of the network, respectively. The set of *transition nodes* $S_{\mathcal{A}}$ includes vertices that represent the interval transitions of the *IMDP* $\mathcal{M}$. More precisely, each transition labelled by $a$ enabled at state $t$ is represented by a transition node $t_a \in S_{\mathcal{A}}$. The set $S_{\mathcal{R}}$ is a copy of the state set $S$ that is used to represent the states reached after having performed the transition; for such states, we connect them to the equivalence class they belong to so to verify the condition of the lifting. The network construction can be seen as an adaptation to the strong case of flow networks used in [21,44].

We take advantage of the above transformation of the "*IMDP* into a network graph" to generate an optimization problem. To this aim, we adopt the same notation of the network optimization setting so we use $f_{u,v}$ to show the "flow" through the arc from $u$ to $v$. In formulating the optimization problem, we use in addition the so-called *balancing constraints* [44] in order to reflect the probabilistic choices in the given *IMDP* $\mathcal{M}$ and to ensure the correct splitting of outgoing flows from the transition nodes in the set $S_{\mathcal{A}}$.

**Definition 11.** *The optimization problem associated to the network* $G(t,\mathcal{R}) = (V,E)$ *is defined as follows:*

$$
\begin{array}{ll}
\mathrm{Min}_f \quad 0 & \\
\textit{subject to} & \\
f_{u,v} \geq 0 & \textit{for each } (u,v) \in E \\
f_{\triangle,t} = 1 & \\
f_{\mathcal{C},\blacktriangledown} = \mu(\mathcal{C}) & \textit{for each } \mathcal{C} \in S/\mathcal{R} \\
\sum_{\{\, u \in V \mid (u,v) \in E \,\}} f_{u,v} - \sum_{\{\, w \in V \mid (v,w) \in E \,\}} f_{v,w} = 0 & \textit{for each } v \in V \setminus \{\triangle, \blacktriangledown\} \\
f_{t_a, v_{\mathcal{R}}} - p_{a,v} \cdot f_{t,t_a} = 0 & \textit{for each } a \in \mathcal{A}(t) \textit{ and } v \in S \\
p_{a,v} \in I(t,a,v) & \textit{for each } a \in \mathcal{A}(t) \textit{ and } v \in S
\end{array}
$$

It is not difficult to see that the optimization problem just defined is not an LP problem, as there are quadratic constraints where the flow variable $f_{t,t_a}$ is multiplied with the "probability" variable $p_{a,v}$. As a matter of fact, for a given $a \in \mathcal{A}(t)$, the variables $p_{a,v}$ have to lie in the interval defined by the interval transition $I(t,a,v)$ and they have to induce a probability distribution, i.e., $p_{a,v} \geq 0$ for each $v \in S$ and $\sum_{v \in S} p_{a,v} = 1$. The non-negativity of the variables comes for free from the constraints $p_{a,v} \in I(t,a,v)$ since $I(t,a,v) \subseteq [0,1]$; $\sum_{v \in S} p_{a,v} = 1$ follows by the flow conservation constrain $\sum_{\{\, u \in V \mid (u,v) \in E \,\}} f_{u,v} - \sum_{\{\, w \in V \mid (v,w) \in E \,\}} f_{v,w} = 0$ for $v = t_a$. Therefore, the optimization problem can be easily cast as an

LP problem by replacing the pair of constraints $f_{t_a,v_\mathcal{R}} - p_{a,v} \cdot f_{t,t_a} = 0$ and $p_{a,v} \in I(t,a,v)$ with the pair of constraints $f_{t_a,v_\mathcal{R}} - \inf I(t,a,v) \cdot f_{t,t_a} \geq 0$ and $f_{t_a,v_\mathcal{R}} - \sup I(t,a,v) \cdot f_{t,t_a} \leq 0$, i.e., the state $v$ is reached from $t$ with probability $p_{a,v} = \frac{f_{t_a,v_\mathcal{R}}}{f_{t,t_a}}$ at least $\inf I(t,a,v)$ and at most $\sup I(t,a,v)$, as required. Taking this modification into account, we can reformulate the optimization problem in Def. 11 as the following LP problem.

**Definition 12 (The $LP(t,\mu,\mathcal{R})$ LP problem).** *The $LP(t,\mu,\mathcal{R})$ LP problem associated to the network graph $G(t,\mathcal{R}) = (V,E)$ is defined as follows:*

$$
\begin{array}{ll}
\text{Min}_f \quad 0 & \\
\textit{subject to} & \\
f_{u,v} \geq 0 & \textit{for each } (u,v) \in E \\
f_{\triangle,t} = 1 & \\
f_{\mathcal{C},\blacktriangledown} = \mu(\mathcal{C}) & \textit{for each } \mathcal{C} \in S/\mathcal{R} \\
\sum_{\{\, u \in V \mid (u,v) \in E \,\}} f_{u,v} - \sum_{\{\, w \in V \mid (v,w) \in E \,\}} f_{v,w} = 0 & \textit{for each } v \in V \setminus \{\triangle, \blacktriangledown\} \\
f_{t_a,v_\mathcal{R}} - \inf I(t,a,v) \cdot f_{t,t_a} \geq 0 & \textit{for each } a \in \mathcal{A}(t) \textit{ and } v \in S \\
f_{t_a,v_\mathcal{R}} - \sup I(t,a,v) \cdot f_{t,t_a} \leq 0 & \textit{for each } a \in \mathcal{A}(t) \textit{ and } v \in S
\end{array}
$$

The feasibility of the resulting LP problem can be seen as an oracle to verify or refute the existence of a probabilistic transition $t \longrightarrow \mu_t$. Formally,

**Lemma 13.** *Given an IMDP $\mathcal{M}$, $t \in S$, $\mu \in \Delta(S)$, and an equivalence relation $\mathcal{R}$ on $S$, the $LP(t,\mu,\mathcal{R})$ LP problem has a feasible solution if and only if there exist $\sigma \in \mathfrak{S}$ and $\nu \in \mathfrak{N}$ inducing $t \longrightarrow \mu_t$ such that $\mu \; \mathcal{L}(\mathcal{R}) \; \mu_t$.*

It is worthwhile to be noted that the resulting scheduler and nature are history-independent, i.e., they base their choice only on the current state (and action, for nature). Moreover, solving the generated LP problem from Def. 11 can be done in polynomial time [33, 34]. The polynomial time complexity, however, is not preserved when uncertainty affects transition probabilities in the model. In fact, in presence of uncertainty, the step condition needs to be checked for any realization of the probability distribution $\mu_s \in \mathcal{P}^{s,a}_\mathcal{R}$. This fact is essentially the main barrier in designing efficient algorithms for probabilistic bisimulation on such uncertain systems which particularly leads the problem to be intractable. To this end, we first model the probabilistic bisimulation problem as the ARC of the uncertain $LP(t,\mu,\mathcal{R})$ LP problem in which the uncertain data is the probability distribution $\mu$. More precisely, by Lem. 13, we can replace in Def. 10 the matching transition $\mu_t \in \mathcal{P}^t_\mathcal{R}$ for $\mu_s \in \mathcal{P}^{s,a}_\mathcal{R}$ such that $\mu_s \; \mathcal{L}(\mathcal{R}) \; \mu_t$ with the check for feasibility of $LP(t,\mu_s,\mathcal{R})$.

Modelling this probabilistic bisimulation game as ARC of an uncertain LP allows the adjustable flow variables $f_{i,j}$ in the $LP(t,\mu,\mathcal{R})$ LP problem to tune themselves to the uncertain probability distribution $\mu$. However, the ARC is in general computationally hard. On the other hand, restricting the adjustable flow variables $f_{i,j}$ to be affinely dependent on the uncertain probability distributions $\mu$ allows us to model the bisimulation problem as *affinely adjustable robust counterpart* of an uncertain LP problem and thus to arrive at a polynomial time

$\mathrm{Min}_{l,w} \quad 0$
subject to
$l_{u,v} + \sum_{k=1}^{n} w^k \cdot \mu(\mathcal{C}_k) \geq 0$ for each $(u,v) \in E$
$l_{\triangle,t} + \sum_{k=1}^{n} w^k \cdot \mu(\mathcal{C}_k) = 1$
$l_{\mathcal{C},\blacktriangledown} + \sum_{k=1}^{n} w^k \cdot \mu(\mathcal{C}_k) = \mu(\mathcal{C}_i)$ for each $\mathcal{C}_i \in S/\mathcal{R}, i = 1, \ldots, n$
$\sum_{\{\, u \mid (u,v) \in E \,\}} (l_{u,v} + \sum_{k=1}^{n} w^k \cdot \mu(\mathcal{C}_k)) - \sum_{\{\, u \mid (v,u) \in E \,\}} (l_{v,u} + \sum_{k=1}^{n} w^k \cdot \mu(\mathcal{C}_k)) = 0$
for each $v \in V \setminus \{\triangle, \blacktriangledown\}$
$l_{t_a,v_{\mathcal{R}}} + \sum_{k=1}^{n} w^k \cdot \mu(\mathcal{C}_k) - \inf I(t,a,v) \cdot (l_{t,t_a} + \sum_{k=1}^{n} w^k \cdot \mu(\mathcal{C}_k)) \geq 0$
for each $a \in \mathcal{A}(t)$ and $v \in S$
$l_{t_a,v_{\mathcal{R}}} + \sum_{k=1}^{n} w^k \cdot \mu(\mathcal{C}_k) - \sup I(t,a,v) \cdot (l_{t,t_a} + \sum_{k=1}^{n} w^k \cdot \mu(\mathcal{C}_k)) \leq 0$
for each $a \in \mathcal{A}(t)$ and $v \in S$
$\forall \mu = (\mu(\mathcal{C}_1), \ldots, \mu(\mathcal{C}_n)) \in \mathcal{P}_{\mathcal{R}}^{s,a}$

**Fig. 1.** Affinely adjustable robust counterpart of the ULP $\{LP(t,\mu,\mathcal{R})\}_{\mu \in \mathcal{P}_{\mathcal{R}}^{s,a}}$.

algorithm to compute the equivalence relation $\mathcal{R}$. From the game semantics viewpoint, such affine dependency restriction reduces the power of the defender to match the challenger's choices and therefore, it leads to a finer (sub-optimal) equivalence relation.

### 4.3 Affinely Adjustable Robust Counterpart for Probabilistic Bisimulation

In this section, we adapt the ARC theory presented in Sec. 2.2 to the setting of probabilistic bisimulation by imposing a restriction on adjustable flow variables $f_{i,j}$ to tune themselves *affinely* upon the uncertain probability distribution $\mu$ in the challenger's uncertainty set $\mathcal{P}_{\mathcal{R}}^{s,a}$. Without loss of generality, we let $\mathcal{C}_1, \ldots, \mathcal{C}_n$ be the equivalence classes induced by $\mathcal{R}$. We encode the *affine dependence* in the network graph $G(t,\mathcal{R}) = (V,E)$ by restricting, for each arch $(i,j) \in E$, the flow variable $f_{i,j}$ to be

$$f_{i,j} = l_{i,j} + \sum_{k=1}^{n} w^k \cdot \mu(\mathcal{C}_k),$$

where the new optimization variables are considered in the vector $l$ and the matrix $W$. Plugging affine equivalences of flow variables, we end up with the affinely adjustable robust counterpart (AARC) of the ULP problem $\{LP(t,\mu,\mathcal{R})\}_{\mu \in \mathcal{P}_{\mathcal{R}}^{s,a}}$ shown in Fig. 1.

In order to show the computational tractability of the AARC, we need to ensure that the uncertainty set $\mathcal{P}_{\mathcal{R}}^{s,a}$ is itself computationally tractable. Formally, a set $\mathcal{P}_{\mathcal{R}}^{s,a}$ is *computationally tractable* [25] if for any vector $\mu$, there is a tractable "separation oracle" that either decides correctly $\mu \in \mathcal{P}_{\mathcal{R}}^{s,a}$ or otherwise, generates a *separator*, i.e., a non-zero vector $r$ such that $r^T\mu \geq \max_{\gamma \in \mathcal{P}_{\mathcal{R}}^{s,a}} r^T\gamma$.

**Proposition 14.** *For every state $s \in S$, action $a \in \mathcal{A}(s)$ and equivalence relation $\mathcal{R}$, the polytopic uncertainty set $\mathcal{P}_{\mathcal{R}}^{s,a}$ is computationally tractable.*

Computational tractability of the polytopic uncertainty sets concludes immediately tractability of the AARC. Formally,

**Theorem 15.** *Given the fixed recourse ULP problem* $\{LP(t,\mu,\mathcal{R})\}_{\mu\in\mathcal{P}^{s,a}_{\mathcal{R}}}$*, the AARC of* $\{LP(t,\mu,\mathcal{R})\}_{\mu\in\mathcal{P}^{s,a}_{\mathcal{R}}}$ *is computationally tractable.*

It is not difficult to see that in the setting of probabilistic bisimulation, the polytopic uncertainty sets $\mathcal{P}^{s}_{\mathcal{R}}$ are closed, convex, and *well structured*, i.e., they can be described by a list of linear inequalities. Thus in our setting, the resulting AARC is also well structured and thus can be solved using highly efficient LP solvers (for instance, CPLEX [2] and Gurobi [1]) even for large-scale cases.

**Theorem 16.** *Given the fixed recourse ULP problem* $\{LP(t,\mu,\mathcal{R})\}_{\mu\in\mathcal{P}^{s,a}_{\mathcal{R}}}$*, the AARC of* $\{LP(t,\mu,\mathcal{R})\}_{\mu\in\mathcal{P}^{s,a}_{\mathcal{R}}}$ *is equivalent to an explicit LP program.*

The "affine decision rules" used to derive the AARC counterpart of the probabilistic bisimulation problem allow us to compute a sub-optimal (finer) probabilistic bisimulation defined as follows.

**Definition 17.** *Let* $\mathcal{R}\subseteq S\times S$ *be an equivalence relation. We say that* $\mathcal{R}$ *is an* AARC probabilistic bisimulation *if* $(s,t)\in\mathcal{R}$ *implies that* $L(s)=L(t)$ *and for each* $a\in\mathcal{A}(s)$*, the AARC of the ULP problem* $\{LP(t,\mu,\mathcal{R})\}_{\mu\in\mathcal{P}^{s,a}_{\mathcal{R}}}$ *is feasible.*

*Furthermore, we write* $s\sim_{AARC} t$ *if there exists an AARC probabilistic bisimulation* $\mathcal{R}$ *such that* $(s,t)\in\mathcal{R}$*.*

An immediate result relating $\sim_{AARC}$ and $\sim_c$ is that the former is a refinement of the latter, as formalized by the following proposition.

**Proposition 18.** *Given* $\mathcal{M}$*, if* $s\sim_{AARC} t$*, then* $s\sim_c t$*, i.e.,* $\sim_{AARC}\subseteq\sim_c$*.*

## 5 Decision Algorithm

In this section, we give a polynomial algorithm computing the probabilistic bisimulation $\sim_{AARC}$. The general idea of the algorithm follows the one of the algorithm in [27] and involves the construction of the polytopes of the challenger's probability distributions. In order to compute $\sim_{AARC}$ an *IMDP* $\mathcal{M}=(S,\bar{s},\mathcal{A},\mathtt{AP},L,I)$, we follow the usual partition refinement approach [12,21, 32,39,44], formalized by the BISIMULATION procedure in Algorithm 1. Namely, we start with $\mathcal{R}$ being the equivalence relation containing the pairs of states with the same labels; then we iteratively refine $\mathcal{R}$ by splitting the states that violate the definition of bisimulation with respect to $\mathcal{R}$. The core part is to check whether two states "violate the definition of bisimulation". This is where our algorithm differs from the one proposed in [27].

The violation is checked by the procedure VIOLATE. We show that this amounts in solving the AARC of the uncertain LP problem $\{LP(t,\mu,\mathcal{R})\}_{\mu\in\mathcal{P}^{s,a}_{\mathcal{R}}}$ as follows. Recall that for $s\in S$ and an action $a\in\mathcal{A}(s)$, we denote by $\mathcal{P}^{s,a}_{\mathcal{R}}$ the polytope of feasible successor distributions over *equivalence classes* of $\mathcal{R}$ with

BISIMULATION($\mathcal{M}$)

1: $\mathcal{R} \leftarrow \{ (s,t) \in S \times S \mid L(s) = L(t) \}$;
2: **repeat**
3: $\quad \mathcal{R}' \leftarrow \mathcal{R}$;
4: $\quad$ **for all** $s \in S$ **do**
5: $\quad\quad D \leftarrow \emptyset$;
6: $\quad\quad$ **for all** $t \in [s]_{\mathcal{R}}$ **do**
7: $\quad\quad\quad$ **for all** $a \in \mathcal{A}(s)$ **do**
8: $\quad\quad\quad\quad$ **if** VIOLATE$(t, \mathcal{R}, \mathcal{P}_{\mathcal{R}}^{s,a})$
9: $\quad\quad\quad\quad\quad D \leftarrow D \cup \{t\}$;
10: $\quad\quad$ split $[s]_{\mathcal{R}}$ in $\mathcal{R}$ into $D$ and $[s]_{\mathcal{R}} \setminus D$;
11: **until** $\mathcal{R} = \mathcal{R}'$;
12: **return** $\mathcal{R}$;

VIOLATE$(t, \mathcal{R}, \mathcal{P}_{\mathcal{R}}^{s,a})$

1: Construct the AARC of the ULP $\{LP(t, \mu, \mathcal{R})\}_{\mu \in \mathcal{P}_{\mathcal{R}}^{s,a}}$ defined in Fig. 1
2: **return** is AARC not feasible?

Algorithm 1: Probabilistic bisimulation algorithm for *IMDP*s

respect to taking the action $a$ in the state $s$, as discussed in Sec. 3. 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 $\mathcal{C}$. As specified in the procedure VIOLATE, we show that it suffices to check the feasibility of the resulting AARC of the constructed uncertain LP problem.

Given an *IMDP* $\mathcal{M}$, let $N = \max\{|S|, |A|\}$. It is not difficult to see that the procedure VIOLATE is called at most $N^4$ times. In every call to this procedure, we need to generate and solve the explicit form of the AARC which is an LP according to Thm. 16, solvable in polynomial time $\mathcal{O}(poly(N))$ (see, e.g., [25,33]). This means that computing $\sim_{AARC}$ can be done in time $|\mathcal{M}|^{\mathcal{O}(1)} \cdot \mathcal{O}(poly(N))$.

**Theorem 19.** *Algorithm 1 computes* $\sim_{AARC}$ *in time polynomial in* $|\mathcal{M}|$.

## 6 Case Studies

We have written a prototypical implementation for computing the bisimulation presented in this paper. Our tool reads a model specification in the input language of the probabilistic model checker PRISM [36] (extended to support also intervals in the transitions), and constructs an explicit-state representation of the state space. Afterwards, it computes the quotient using Algorithm 1.

Table 1 shows the performance of our prototype on a number of case studies taken from the PRISM website [4], where we have relaxed some of the probabilistic choices to intervals. The machine we used for the experiments is a 3.6 GHz Intel Core i7-4790 with 16 GB 1600 MHz DDR3 RAM of which 12GB were assigned to the tool. Despite using an explicit representation for the model, the prototype is able to manage cases studies in the order of millions of states and transitions (columns "Model", "$|S_i|$", and "$|I_i|$"). The time in seconds required to compute the bisimulation relation and the corresponding quotient *IMDP*, shown in columns "$t_\sim$", "$|S_\sim|$", and "$|I_\sim|$", is much less than the time expected from the theoretical analysis of the algorithm: this is motivated by the fact that we

**Table 1.** Experimental evaluation of the bisimulation computation

| Model | $\lvert S_i\rvert$ | $\lvert I_i\rvert$ | $S/L$ | $t_\sim$ (s) | $\lvert S_\sim\rvert$ | $\lvert I_\sim\rvert$ |
|---|---|---|---|---|---|---|
| Consensus-Shared-Coin-3 | 5 216 | 13 380 | 2 | 0 | 787 | 1 770 |
| Consensus-Shared-Coin-4 | 43 136 | 144 352 | 2 | 2 | 2 189 | 5 621 |
| Consensus-Shared-Coin-5 | 327 936 | 1 363 120 | 2 | 23 | 5 025 | 14 192 |
| Consensus-Shared-Coin-6 | 2 376 448 | 11 835 456 | 2 | 219 | 10 173 | 30 861 |
| Crowds-5-10 | 111 294 | 261 444 | 2 | 1 | 107 | 153 |
| Crowds-5-20 | 2 061 951 | 7 374 951 | 2 | 17 | 107 | 153 |
| Crowds-5-30 | 12 816 233 | 61 511 033 | 2 | 116 | 107 | 153 |
| Crowds-5-40 | 44 045 030 | 266 812 421 | 2 | 464 | 125 | 198 |
| Mutual-Exclusion-PZ-3 | 3 008 | 10 868 | 2 | 0 | 1 123 | 3 939 |
| Mutual-Exclusion-PZ-4 | 48 128 | 231 040 | 2 | 0 | 7 319 | 32 630 |
| Mutual-Exclusion-PZ-5 | 770 048 | 4 611 072 | 2 | 7 | 32 053 | 168 151 |
| Mutual-Exclusion-PZ-6 | 3 377 344 | 25 470 144 | 2 | 98 | 109 986 | 649 360 |
| Dining-Phils-LR-nofair-4 | 9 440 | 40 120 | 4 | 0 | 1 232 | 5 037 |
| Dining-Phils-LR-nofair-5 | 93 068 | 494 420 | 4 | 1 | 9 408 | 49 467 |
| Dining-Phils-LR-nofair-6 | 917 424 | 5 848 524 | 4 | 14 | 76 925 | 487 620 |
| Dining-Phils-LR-nofair-7 | 9 043 420 | 67 259 808 | 4 | 173 | 646 928 | 4 804 695 |

have implemented optimizations, such as caching equivalent LP problems, which improve the runtime of our algorithm in practice. Because of this, we never had to solve more than 30 LP problems in a single tool run, thereby avoiding the potentially costly solution of LP problems from becoming a bottleneck.

*Acknowledgments.* We would like to thank Arkadi Nemirovski (Georgia Institute of Technology) and Daniel Kuhn (EPFL) for many invaluable and insightful discussions. This work is supported by the EU 7th Framework Programme under grant agreements 295261 (MEALS) and 318490 (SENSATION), by the DFG as part of SFB/TR 14 AVACS, by the ERC Advanced Investigators Grant 695614 (POWVER), by the CAS/SAFEA International Partnership Program for Creative Research Teams, by the National Natural Science Foundation of China (Grants No. 61472473, 61532019, 61550110249, 61550110506), by the Chinese Academy of Sciences Fellowship for International Young Scientists, and by the CDZ project CAP (GZ 1023).

## References

1. Gurobi 4.0.2, `http://www.gurobi.com/`
2. IBM ILOG CPLEX Optimizer, `http://www.ibm.com/software/commerce/optimization/cplex-optimizer/`
3. PICOS: A Python interface for conic optimization solvers, `http://picos.zib.de/`
4. PRISM model checker, `http://www.prismmodelchecker.org/`
5. Abate, A., El Ghaoui, L.: Robust model predictive control through adjustable variables: an application to path planning. In: CDC. pp. 2485–2490 (2004)

6. Ben-Tal, A., El Ghaoui, L., Nemirovski, A.: Robust optimization. Princeton University Press (2009)
7. Ben-Tal, A., Goryashko, A., Guslitzer, E., Nemirovski, A.: Adjustable robust solutions of uncertain linear programs. Math. Programming 99(2), 351–376 (2004)
8. Ben-Tal, A., Nemirovski, A.: Robust solutions of uncertain linear programs. Operations research letters 25, 1–13 (1999)
9. Benedikt, M., Lenhardt, R., Worrell, J.: LTL model checking of interval Markov chains. In: TACAS. LNCS, vol. 7795, pp. 32–46 (2013)
10. Bertsimas, D., Brown, D.B., Caramanis, C.: Theory and applications of robust optimization. SIAM review 53(3), 464–501 (2011)
11. Billingsley, P.: Probability and Measure. John Wiley and Sons (1979)
12. Cattani, S., Segala, R.: Decision algorithms for probabilistic bisimulation. In: CONCUR. LNCS, vol. 2421, pp. 371–385 (2002)
13. Cattani, S., Segala, R., Kwiatkowska, M., Norman, G.: Stochastic transition systems for continuous state spaces and non-determinism. In: FoSSaCS. LNCS, vol. 3441, pp. 125–139 (2005)
14. Chatterjee, K., Sen, K., Henzinger, T.A.: Model-checking omega-regular properties of interval Markov chains. In: FoSSaCS. LNCS, vol. 4962, pp. 302–317 (2008)
15. Chen, T., Han, T., Kwiatkowska, M.: On the complexity of model checking interval-valued discrete time Markov chains. Inf. Process. Lett. 113(7), 210–216 (2013)
16. Coste, N., Hermanns, H., Lantreibecq, E., Serwe, W.: Towards performance prediction of compositional models in industrial GALS designs. In: CAV. LNCS, vol. 5643, pp. 204–218 (2009)
17. Dantzig, G.B., Madansky, A.: On the solution of two-stage linear programs under uncertainty. In: Proc. Fourth Berkeley Symp. on Math. Statist. and Prob., Vol. 1. pp. 165–176 (1961)
18. Delahaye, B., Katoen, J.P., Larsen, K.G., Legay, A., Pedersen, M.L., Sher, F., Wasowski, A.: New results on abstract probabilistic automata. In: ACSD. pp. 118–127 (2011)
19. Delahaye, B., Larsen, K.G., Legay, A., Pedersen, M.L., Wasowski, A.: Decision problems for interval Markov chains. In: LATA. LNCS, vol. 6638, pp. 274–285 (2011)
20. Fecher, H., Leucker, M., Wolf, V.: Don't know in probabilistic systems. In: SPIN. LNCS, vol. 3925, pp. 71–88 (2006)
21. Ferrer Fioriti, L.M., Hashemi, V., Hermanns, H., Turrini, A.: Deciding probabilistic automata weak bisimulation: Theory and practice. FAoC 28(1), 109–143 (2016)
22. Gebler, D., Hashemi, V., Turrini, A.: Computing behavioral relations for probabilistic concurrent systems. In: Stochastic Model Checking. LNCS, vol. 8453, pp. 117–155 (2014)
23. Givan, R., Leach, S.M., Dean, T.L.: Bounded-parameter Markov decision processes. Artif. Intell. 122(1-2), 71–109 (2000)
24. Goerigk, M.: ROPI–a robust optimization programming interface for C++. Optimization Methods and Software 29(6), 1261–1280 (2014)
25. Grötschel, M., Lovász, L., Schrijver, A.: The ellipsoid method and its consequences in combinatorial optimization. Combinatorica 1(2), 169–197 (1981)
26. Hahn, E.M., Han, T., Zhang, L.: Synthesis for PCTL in parametric Markov decision processes. In: NASA Formal Methods. LNCS, vol. 6617, pp. 146–161 (2011)
27. Hashemi, V., Hatefi, H., Krčál, J.: Probabilistic bisimulations for PCTL model checking of interval MDPs (extended version). In: SynCoP. EPTCS, vol. 145, pp. 19–33 (2014)

28. Hashemi, V., Hermanns, H., Song, L., Subramani, K., Turrini, A., Wojciechowski, P.: Compositional bisimulation minimization for interval Markov decision processes. In: LATA. LNCS, vol. 9618, pp. 114–126 (2016)
29. Hermanns, H., Katoen, J.P.: Automated compositional Markov chain generation for a plain-old telephone system. SCP 36(1), 97–127 (2000)
30. Iyengar, G.N.: Robust dynamic programming. Mathematics of Operations Research 30(2), 257–280 (2005)
31. Jonsson, B., Larsen, K.G.: Specification and refinement of probabilistic processes. In: LICS. pp. 266–277 (1991)
32. Kanellakis, P.C., Smolka, S.A.: CCS expressions, finite state processes, and three problems of equivalence. I&C 86(1), 43–68 (1990)
33. Karmarkar, N.: A new polynomial-time algorithm for linear programming. Combinatorica 4(4), 373–395 (1984)
34. Khachyan, L.G.: A polynomial algorithm in linear programming. Soviet Mathematics Doklady 20(1), 191–194 (1979)
35. Kozine, I., Utkin, L.V.: Interval-valued finite Markov chains. Reliable Computing 8(2), 97–113 (2002)
36. Kwiatkowska, M., Norman, G., Parker, D.: PRISM 4.0: Verification of probabilistic real-time systems. In: CAV. LNCS, vol. 6806, pp. 585–591 (2011)
37. Löfberg, J.: Automatic robust convex programming. Optimization methods and software 17(1), 115–129 (2012)
38. Nilim, A., El Ghaoui, L.: Robust control of Markov decision processes with uncertain transition matrices. Operations Research 53(5), 780–798 (2005)
39. Paige, R., Tarjan, R.E.: Three partition refinement algorithms. SIAM J. on Computing 16(6), 973–989 (1987)
40. Puggelli, A.: Formal Techniques for the Verification and Optimal Control of Probabilistic Systems in the Presence of Modeling Uncertainties. Ph.D. thesis, University of California, Berkeley (2014)
41. Puggelli, A., Li, W., Sangiovanni-Vincentelli, A.L., Seshia, S.A.: Polynomial-time verification of PCTL properties of MDPs with convex uncertainties. In: CAV. LNCS, vol. 8044, pp. 527–542 (2013)
42. Segala, R.: Modeling and Verification of Randomized Distributed Real-Time Systems. Ph.D. thesis, MIT (1995)
43. Sen, K., Viswanathan, M., Agha, G.: Model-checking Markov chains in the presence of uncertainties. In: TACAS. LNCS, vol. 3920, pp. 394–410 (2006)
44. Turrini, A., Hermanns, H.: Polynomial time decision algorithms for probabilistic automata. I&C 244, 134–171 (2015)
45. Wolff, E.M., Topcu, U., Murray, R.M.: Robust control of uncertain Markov decision processes with temporal logic specifications. In: CDC. pp. 3372–3379 (2012)
46. Wu, D., Koutsoukos, X.D.: Reachability analysis of uncertain systems using bounded-parameter Markov decision processes. Artif. Intell. 172(8-9), 945–954 (2008)
47. Yi, W.: Algebraic reasoning for real-time probabilistic processes with uncertain information. In: FTRTFT. LNCS, vol. 863, pp. 680–693 (1994)