Model checking for imprecise Markov chains

Matthias C. M. Troffaes
Durham University, UK
matthias.troffaes@gmail.com

Damjan Škulj
University of Ljubljana, Slovenia
damjan.skulj@fdv.uni-lj.si

Abstract

We extend probabilistic computational tree logic for expressing properties of Markov chains to imprecise Markov chains, and provide an efficient algorithm for model checking of imprecise Markov chains. Thereby, we provide a formal framework to answer a very wide range of questions about imprecise Markov chains, in a systematic and computationally efficient way, whilst at the same time improving and simplifying model checking for a fairly broad class of Markov decision processes.

Keywords. imprecise Markov chain, model checking, parse tree, logic, computation

1 Introduction

In a nutshell, model checking [2] is a model-based technique which automates the verification of the reliability of a system. To do so, firstly we need a specification of the *system model*, and secondly a specification of *system properties*. The system is then deemed reliable if the model satisfies those properties. Model checking can be performed manually, for instance through peer review, however, for larger models, this can take a lot of time and can cost a lot of money. Moreover, peer review typically does not catch all errors. Therefore, there is a strong need for automating the model checking process.

Interestingly, in model checking literature, a lot of work has been done for so-called *Markov decision processes*—these are Markov chains where the transition probabilities depend on non-deterministic choices, about which we are completely vacuous, and so in fact they are quite closely related to imprecise probability [4, 13, 14, 11, 12].[1] Indeed, so-called *imprecise Markov chains* have been studied by many authors. Hartfiel [7, 6] proposed Markov set-chains models, where transition matrices form matrix intervals. He does not connect his work with imprecise probability formally, but uses several methods similar to those developed in imprecise probability theory. Another more recent case where interval probabilities are involved in the study of Markov chains, outside the formal theory of imprecise probability, is described in [8]. In that work, the interval probabilities are formed by abstraction of a precise Markov chain, that is by merging states. Also this model could be seen as an imprecise Markov chain. A more formal connection between Markov chains and interval probabilities was established by Kozine and Utkin [9], and generalised by Škulj [10] and De Cooman et al. [5]. In [5], lower and upper expectation operators are used instead of sets of probabilities, leading to simpler calculations and more elegant proofs. We follow their approach as well.

In this paper, we investigate model checking techniques for imprecise Markov chains. Although, theoretically, these models can already be checked using existing techniques for Markov decision processes [2, Sec. 10.6] [1], in this paper we follow [5] and restrict ourselves to a very special type of Markov decision process, namely those imprecise Markov chains for which bounds on transition probabilities can be calculated simply by means of linear programming.

Indeed, imprecise Markov chains can be formally connected to Markov decision processes as follows: the extreme points of the set of transition probabilities in the imprecise Markov chain correspond to the set of actions in the Markov decision process. Existing techniques for Markov decision processes require the set of actions to be finite. Interestingly, in our approach, the set of extreme points is not required to be finite, as our model checking algorithm does not depend on the cardinality of the set of extreme points. A second advantage of our approach is that we express our algorithm directly in terms of constraints (lower and upper expectations), rather than in terms of extreme

[1] In operations research, the term Markov decision process has an entirely different meaning.

points or, actions, if you like. This is computationally much more efficient than model checking algorithms which work with actions directly, because it is well known that the number of extreme points is usually much greater than the number of constraints required to describe the same set (for example, see [3]). Even in those cases where the number of constraints is larger, one can still use the extreme points to calculate lower and upper expectations, whence our approach never does worse than existing algorithms which use actions directly.

From the model checking perspective, this paper contributes algorithms that are potentially much faster than traditional methods for Markov decision processes, in essence because we can circumvent sets of probabilities, and instead focus on the constraints. The algorithm is also simpler, and resembles much more closely the one for precise Markov chains. From the imprecise probability perspective, the contribution of this paper is the development of a formal framework to answer a very wide range of questions about Markov chains in a computationally efficient way. In doing so, we put various existing results from the literature about imprecise Markov chains into a common framework.

This paper is structured as follows. Section 2 reviews the existing theory of model checking for Markov chains. Section 3 explains how the logic and model checking algorithm can be adapted to suit imprecise Markov chains. Section 4 has an example. We conclude in Section 5.

2 Model Checking for Markov Chains

Before we move on to imprecise Markov chains, in this section, we briefly review the standard model checking framework for Markov chains [2, Chapter 10]. For simplicity, in this paper, we will restrict ourselves to finite state discrete time Markov chains.

2.1 Model Specification: Transitions, Labelling, Paths, Probabilities

Definition 1 (Markov Chain) *A (finite state, discrete time)* Markov chain *consists of:*

- *a finite set of* states S,
- *an* initial probability $\mathrm{P}_0(s)$ *for all* $s \in S$, *and*
- transition probabilities $\mathbf{P}(s,t)$ *for all* $(s,t) \in S^2$.

For specifying properties of Markov chains, it is useful to introduce labels as well:

Definition 2 (Labelling) *Consider a finite set of* atomic propositions AP. *A* labelling *of states is then simply a mapping* $L\colon S \to \wp(AP)$ *which associates a set of atomic propositions with every state.*

An atomic proposition is just a convenient way to specify a subset of states. For example, in a reliability problem, we could have

$$AP = \{\text{system working}, \text{system broken}\}, \tag{1}$$

with states of the Markov chain labelled accordingly. In more advanced problems, it is sometimes convenient to allow each state to carry more than one atomic proposition. A trivial labelling is $L(s) = \{s\}$ for every $s \in S$; this is what we will usually assume, unless otherwise stated.

The *digraph* of a Markov chain is a graph where each state is represented by a vertex, and each possible transition ($\mathbf{P}(s,t) > 0$) is an edge—this is the picture we generally draw for a Markov chain.

A *path* is then simply an infinite sequence of states on the digraph:

$$\pi = s_0 s_1 s_2 \cdots \in S^{\mathbb{N}}. \tag{2}$$

The *trace* of a path is its induced sequence of labels:

$$\operatorname{trace}(s_0 s_1 s_2 \cdots) = L(s_0)L(s_1)L(s_2)\cdots \in \wp(AP)^{\mathbb{N}}. \tag{3}$$

A *cylinder set* is a set of paths with a common prefix:

$$\operatorname{cyl}(s_0 s_1 \cdots s_n) = \{s_0 s_1 \cdots s_n s_{n+1} s_{n+2} \cdots : s_{n+1} s_{n+2} \cdots \in S^{\mathbb{N}}\}. \tag{4}$$

For example, the set of paths starting from state s is the cylinder set

$$\operatorname{cyl}(s) = \{s_0 s_1 s_2 \cdots \in S^{\mathbb{N}} \colon s_0 = s\}. \tag{5}$$

Cylinder sets play a central role in Markov chains because these are the sets for which we can very easily calculate their probability:

$$\Pr(\operatorname{cyl}(s_0 s_1 \cdots s_n)) = \mathrm{P}_0(s_0) \prod_{i=0}^{n-1} \mathbf{P}(s_i, s_{i+1}). \tag{6}$$

Also of interest is the probability of a cylinder set conditional on knowing the initial state s_0:

$$\Pr(\operatorname{cyl}(s_0 s_1 \cdots s_n) \mid s_0) = \prod_{i=0}^{n-1} \mathbf{P}(s_i, s_{i+1}). \tag{7}$$

state formula	meaning
$s \vDash \text{true}$	always satisfied
$s \vDash a$	$a \in L(s)$
$s \vDash \neg\Phi$	not $s \vDash \Phi$
$s \vDash \Phi \wedge \Psi$	$s \vDash \Phi$ and $s \vDash \Psi$
$s \vDash \mathbb{P}_J(\phi)$	$\Pr(s \vDash \phi) \in J$
path formula	**meaning**
$\pi \vDash \bigcirc\Phi$	$\pi[1] \vDash \Phi$
$\pi \vDash \Phi \mathbin{\mathsf{U}} \Psi$	$\exists j \geq 0$: $\big((\forall 0 \leq k < j : \pi[k] \vDash \Phi)$ and $\pi[j] \vDash \Psi\big)$
$\pi \vDash \Phi \mathbin{\mathsf{U}}^{\leq n} \Psi$	$\exists 0 \leq j \leq n$: $\big((\forall 0 \leq k < j : \pi[k] \vDash \Phi)$ and $\pi[j] \vDash \Psi\big)$

Table 1: Semantics of state and path formulas.

2.2 Property Specification: Probabilistic Computation Tree Logic

A formal and very useful way of specifying properties of Markov chains goes via *probabilistic computation tree logic*, where we distinguish between two types of properties:

1. Properties of *states* of the system:

$$s \vDash \Phi \text{ if state } s \text{ satisfies state formula } \Phi. \quad (8)$$

2. Properties of *paths* of the system:

$$\pi \vDash \phi \text{ if path } \pi \text{ satisfies path formula } \phi. \quad (9)$$

State formulas are denoted by upper case greek letters Φ, Ψ, and so on. Path formulas are denoted by lower case greek letters ϕ, ψ, and so on.

State and path formulas Φ and ϕ are taken from a grammar with the following syntax:

$$\Phi ::= \text{true} \quad | \quad a \quad | \quad \Phi_1 \wedge \Phi_2 \quad | \quad \neg\Phi \quad | \quad \mathbb{P}_J(\phi) \quad (10)$$

$$\phi ::= \bigcirc\Phi \quad | \quad \Phi \mathbin{\mathsf{U}} \Psi \quad | \quad \Phi \mathbin{\mathsf{U}}^{\leq n} \Psi \quad (11)$$

where $\bigcirc$ ("next") and U ("until") operators must be directly preceded by a $\mathbb{P}_J$ operator. The semantics, or meaning, of these formulas is summarized in Table 1, where

$$\Pr(s \vDash \phi) = \Pr(\{\pi \in \text{cyl}(s) : \pi \vDash \phi\} \mid s) \quad (12)$$

and $\pi[i] = s_i$ for $\pi = s_0 s_1 \cdots$. The usual operators can be derived from the above ones:

$$\Phi \vee \Psi := \neg((\neg\Phi) \wedge (\neg\Psi)) \qquad \Phi \text{ or } \Psi \quad (13)$$

$$\Diamond\Phi := \text{true} \mathbin{\mathsf{U}} \Phi \qquad \text{eventually } \Phi \quad (14)$$

$$\Box\Phi := \neg(\Diamond(\neg\Phi)) \qquad \text{always } \Phi \quad (15)$$

as well as bounded versions $\Diamond^{\leq n}$ and $\Box^{\leq n}$ of $\Diamond$ and $\Box$, implication, exclusive or, equivalence, and so on.

For a non-trivial example of a state formula, consider a system modelled by a Markov chain whose states are labelled with atomic propositions taken from $AP = \{\text{working}, \text{broken}\} = \{w, b\}$. The property

$$s \vDash \mathbb{P}_{[0.99,1]}\Big(\Diamond^{\leq 20}\big(w \wedge \mathbb{P}_{[0,0.01]}(w \mathbin{\mathsf{U}}^{\leq 10} b)\big)\Big) \quad (16)$$

is then satisfied when, starting from state s, with probability at least 0.99, within 20 steps, the system reaches a working state, from which the probability that the system breaks down within the next 10 steps is less than 0.01. Model checking provides an automated method for verifying any such formula.

Before we move on to the algorithm, one technical issue that arises is is whether $\mathbb{P}_J(\phi)$ is well defined, or more specifically, whether the probability $\Pr(s \vDash \phi)$ (see Eq. (12)) exists. The key observation is:

Theorem 1 *For every state s and every path formula ϕ,*

$$\{\pi \in \text{cyl}(s) : \pi \vDash \phi\} \quad (17)$$

is a countable union of cylinder sets.

The above theorem, along with the fact that the probability specification in Eqs. (6) and (7) can be extended to a σ-field containing all countable unions of cylinder sets, imply that $\Pr(s \vDash \phi)$ exists; see for instance [2, Lemma 10.39] for a proof of Theorem 1.

2.3 Model Checking: Automated Algorithm

The central question we aim to answer is: given a state s and a state formula Φ, does s satisfy Φ? In a nutshell, the algorithm works as follows:

- traverse the *parse tree* of Φ, visiting all subformulas, starting at the leaves of the tree and working back to its root,
- at each subformula Ψ, calculate set of states which *satisfy* Ψ

$$\text{Sat}(\Psi) = \{s' \in S : s' \vDash \Psi\}, \quad (18)$$

- check that $s \in \text{Sat}(\Phi)$.

Figure 1 shows the parse tree of the formula on the right hand side of Eq. 16, and Figure 2 demonstrates how we evaluate Sat through the parse tree.

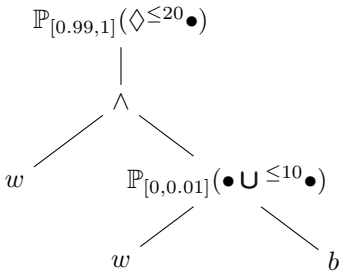

Figure 1: Parse tree of the formula on the right hand side of Eq. 16.

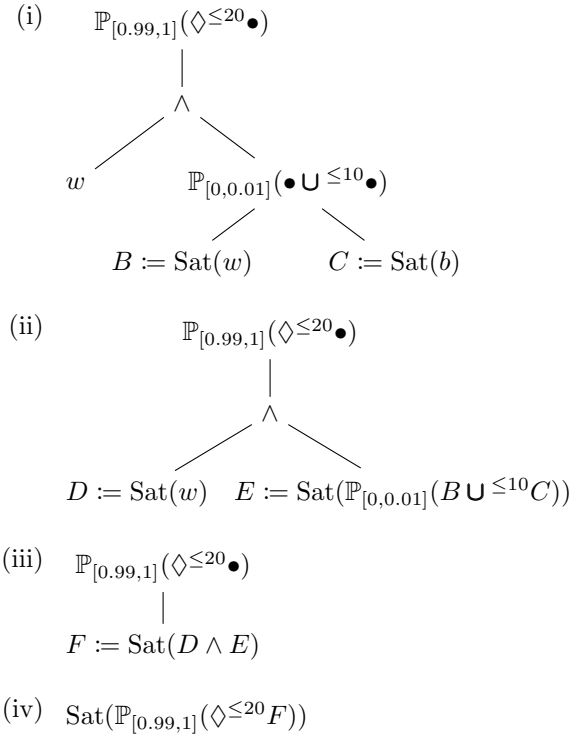

Figure 2: Evaluating Sat through the parse tree.

Effectively, we calculate $\mathrm{Sat}(\Psi)$ by applying the following formulas recursively:

$$\mathrm{Sat}(\mathrm{true}) = S \tag{19}$$
$$\mathrm{Sat}(a) = \{s \in S \colon a \in L(s)\} \tag{20}$$
$$\mathrm{Sat}(\Phi \wedge \Psi) = \mathrm{Sat}(\Phi) \cap \mathrm{Sat}(\Psi) \tag{21}$$
$$\mathrm{Sat}(\neg\Phi) = S \setminus \mathrm{Sat}(\Phi) \tag{22}$$
$$\mathrm{Sat}(\mathbb{P}_J(\phi)) = \{s \in S \colon \Pr(s \vDash \phi) \in J\} \tag{23}$$

where

$$\Pr(s \vDash \bigcirc\Phi) = \mathbf{P}(s, \mathrm{Sat}(\Phi)) = \sum_{t \in \mathrm{Sat}(\Phi)} \mathbf{P}(s,t) \tag{24}$$

and, under certain regularity conditions,

$$\Pr(s \vDash \Phi \mathbin{\mathsf{U}} \Psi) = \lim_{n\to\infty} \Pr(s \vDash \Phi \mathbin{\mathsf{U}}^{\leq n} \Psi). \tag{25}$$

Note that there are efficient ways to determine $\Pr(s \vDash \Phi \mathbin{\mathsf{U}}^{\leq n} \Psi)$ for large n. There are also methods for evaluating $\Pr(s \vDash \Phi \mathbin{\mathsf{U}} \Psi)$ directly, under arbitrary conditions; for details we refer to [2, pp. 761–762].

Thus, the only probability we yet have to evaluate is $\Pr(s \vDash \Phi \mathbin{\mathsf{U}}^{\leq n} \Psi)$. Let $C \coloneqq \mathrm{Sat}(\Phi)$ and $B \coloneqq \mathrm{Sat}(\Psi)$, and for simplicity assume a trivial labelling $L(s) = \{s\}$, so we can write $\Pr(s \vDash C \mathbin{\mathsf{U}}^{\leq n} B)$ for $\Pr(s \vDash \Phi \mathbin{\mathsf{U}}^{\leq n} \Psi)$.

- If $s \in B$ then $\Pr(s \vDash C \mathbin{\mathsf{U}}^{\leq n} B) = 1$.
- Otherwise, if $s \notin C$ then $\Pr(s \vDash C \mathbin{\mathsf{U}}^{\leq n} B) = 0$.
- Otherwise, $s \in C \setminus B$, and

$$\Pr(s \vDash C \mathbin{\mathsf{U}}^{\leq n} B) = \Pr_*(s \vDash \bigcirc^n B) \tag{26}$$
$$= \mathbf{P}_*^n(s, B) \tag{27}$$
$$= \sum_{t \in B} \mathbf{P}_*^n(s,t) \tag{28}$$

where $\Pr_*$ and $\mathbf{P}_*$ denotes the probabilities associated with the modified Markov chain where all states, except those in $C \setminus B$, have been made absorbing. There are many efficient ways to evaluate the n-step transition probability matrix $\mathbf{P}_*^n(s,t)$.

3 Model Checking for Imprecise Markov Chains

3.1 Model Specification: Credal Sets and Upper Transition Operator

Definition 3 (Imprecise Markov Chain) *A (finite state, discrete time)* imprecise Markov chain *consists of:*

- *a finite set of* states S,
- *an* initial credal set $\mathcal{P}_0$ *on* S, *specified through linear constraints on* $\mathrm{P}_0(\cdot)$, *and*
- *a* transition credal set $\boldsymbol{\mathcal{P}}(s)$ *for each* $s \in S$, *specified through linear constraints on* $\mathbf{P}(s,\cdot)$.

The sensitivity interpretation is that, at each step, the transition is described by $\mathbf{P}(s,\cdot) \in \boldsymbol{\mathcal{P}}(s)$ but we do not know which element.

Clearly, this model is not the most general one. Firstly, it has *separately specified rows* [10], that is, a separate model for transitions from each state. Secondly, it features *non-stationarity* (in the sensitivity interpretation), as the actual transition probabilities may change from step to step, and are only constrained to belong to their credal set at each step. These assumptions make the model computationally tractable, and almost as easy to work with as precise Markov chains. Indeed, it turns out that for most calculations of typical interest, we can entirely ignore the credal sets, and instead work with a single operator that can be evaluated through linear programming [5, 10].

Indeed, for typical calculations, we are interested in lower and upper probabilities of events, or more generally, lower and upper expectations of random quantities. Let $\mathcal{L}(S)$ denote the set of all random quantities, also called *gambles*, on S. Gambles are denoted by lower case letters f, g, and so on.

For instance, we could be interested in the lower and upper expectation of a gamble f on the next state, given the current state s:

Definition 4 (Transition Operators) *The operators* $\underline{\mathrm{T}}\colon \mathcal{L}(S) \to \mathcal{L}(S)$ *and* $\overline{\mathrm{T}}\colon \mathcal{L}(S) \to \mathcal{L}(S)$ *defined by*

$$(\underline{\mathrm{T}}(f))(s) \coloneqq \min_{\mathbf{P}(s,\cdot)\in\boldsymbol{\mathcal{P}}(s)} \sum_{t\in S} \mathbf{P}(s,t) f(t), \tag{29}$$

$$(\overline{\mathrm{T}}(f))(s) \coloneqq \max_{\mathbf{P}(s,\cdot)\in\boldsymbol{\mathcal{P}}(s)} \sum_{t\in S} \mathbf{P}(s,t) f(t), \tag{30}$$

are called the lower and upper transition operators.

A key point is that calculation of $\underline{\mathrm{T}}(f)$ and $\overline{\mathrm{T}}(f)$ is efficient. In fact, once we have specified the linear constraints that determine the credal sets $\boldsymbol{\mathcal{P}}(s)$ for each $s \in S$, Eqs. (29) and (30) can be evaluated via linear programming [11, Chapter 3]. Many interesting characteristics of the imprecise Markov chain can be derived just from $\overline{\mathrm{T}}$ [5]. To ease notation, we will often write $\overline{\mathrm{T}}f$ for $\overline{\mathrm{T}}(f)$.

More generally, we might be interested in the lower and upper expectations of gambles on the state after exactly n steps, given the current state s. For instance, what is the upper probability of ending up in $B \subseteq S$ after exactly n steps? Let us use the usual notation for the indicator gamble of a set B:

$$I_B(s) \coloneqq \begin{cases} 1 & \text{if } s \in B, \\ 0 & \text{if } s \notin B. \end{cases} \tag{31}$$

Clearly, $(\overline{\mathrm{T}} I_B)(s)$ is the desired upper probability for $n = 1$. By marginal extension [11, Sec. 6.7.2], it follows that $\overline{\mathrm{T}}(\overline{\mathrm{T}} I_B))(s)$ is the desired upper probability for $n = 2$; we will use the notation $\overline{\mathrm{T}}^2 I_B$ for $\overline{\mathrm{T}}(\overline{\mathrm{T}} I_B)$. Similarly, by $(\overline{\mathrm{T}}^n I_B)(s)$, we denote the desired upper probability for arbitrary n, found by repeated application of $\overline{\mathrm{T}}$.

Definition 5 (n-Step Transition Operators) *The operators* $\underline{\mathrm{T}}^n\colon \mathcal{L}(S) \to \mathcal{L}(S)$ *defined by*

$$(\underline{\mathrm{T}}^n f)(s) = \begin{cases} (\underline{\mathrm{T}}(\underline{\mathrm{T}}^{n-1} f))(s) & \text{if } n > 1 \\ (\underline{\mathrm{T}} f)(s) & \text{if } n = 1 \end{cases} \tag{32}$$

is called the n-step lower transition operator, *and*

$$(\overline{\mathrm{T}}^n f)(s) = \begin{cases} (\overline{\mathrm{T}}(\overline{\mathrm{T}}^{n-1} f))(s) & \text{if } n > 1 \\ (\overline{\mathrm{T}} f)(s) & \text{if } n = 1 \end{cases} \tag{33}$$

is called the n-step upper transition operator.

For model checking, we will use the notation

$$\underline{\mathrm{T}}^n(s, B) \coloneqq \underline{\mathrm{T}}^n(I_B)(s) \tag{34}$$

$$\overline{\mathrm{T}}^n(s, B) \coloneqq \overline{\mathrm{T}}^n(I_B)(s) \tag{35}$$

to denote the lower and upper probability of ending up in B given that we started in s.

3.2 Property Specification: Imprecise Probabilistic Computation Tree Logic

The syntax and semantics of state and path formulas is as before, with only two differences.

First, for simplicity, here, we will exclude U from our logic, to avoid technical issues with countable unions of cylinder sets resulting from the U operator. The $\bigcirc$ and $\mathsf{U}^{\leq n}$ operators pose no problem. Consequently, we can impose an upper bound on the number of steps, after which we are no longer interested in the Markov chain, so all sets and partitions involved can be assumed to have finite cardinality. In this way, we avoid a host of technical problems. Problems involving infinite horizon require additional considerations and will therefore be tackled elsewhere. In any case, for practical applications, time-bounded properties will often be sufficient.

Secondly, and more crucially, we need to use a *different semantics for* $\mathbb{P}$:

$$s \vDash \mathbb{P}_J(\phi) \tag{36}$$

will mean that

$$[\underline{\Pr}(s \vDash \phi), \overline{\Pr}(s \vDash \phi)] \subseteq J \tag{37}$$

where

$$\underline{\Pr}(s \vDash \phi) \coloneqq \underline{\Pr}(\{\pi \in \mathrm{cyl}(s) \colon \pi \vDash \phi\} \mid s) \tag{38}$$

$$\overline{\Pr}(s \vDash \phi) \coloneqq \overline{\Pr}(\{\pi \in \mathrm{cyl}(s) \colon \pi \vDash \phi\} \mid s) \tag{39}$$

and the right hand sides denote the lower and upper expectations corresponding to *natural extension* [13, 14] [11, Sec. 8.1], where the Markov condition is expressed through epistemic irrelevance [5].

3.3 Model Checking: Automated Algorithm

Again, the central question is: given a state s and a state formula Φ, does s satisfy Φ? It is easy to see that we can implement an algorithm exactly as before, namely by *evaluating* Sat *throughout the parse tree.*

The non-trivial differences are:

$$\underline{\Pr}(s \vDash \bigcirc \Phi) = \underline{\mathrm{T}}(s, \mathrm{Sat}(\Phi)) \tag{40}$$

and

$$\underline{\Pr}(s \vDash \Phi \mathbin{\mathsf{U}}^{\leq n} \Psi) = \begin{cases} 1 & \text{if } s \in \mathrm{Sat}(\Psi) \\ 0 & \text{if } s \notin \mathrm{Sat}(\Phi) \cup \mathrm{Sat}(\Psi) \\ \underline{\mathrm{T}}^n_*(s, \mathrm{Sat}(\Psi)) & \text{if } s \in \mathrm{Sat}(\Phi) \setminus \mathrm{Sat}(\Psi) \end{cases} \tag{41}$$

where the $*$ denotes the imprecise Markov chain with all states outside $\mathrm{Sat}(\Phi) \setminus \mathrm{Sat}(\Psi)$ being made absorbing:

$$(\underline{\mathrm{T}}_* f)(s) = \begin{cases} (\underline{\mathrm{T}} f)(s) & \text{if } s \in \mathrm{Sat}(\Phi) \setminus \mathrm{Sat}(\Psi), \\ f(s) & \text{otherwise.} \end{cases} \tag{42}$$

The formulas for upper expectations are trivially similar.

4 Example

Consider a Markov chain with the set of states $S = \{s_1, s_2, s_3, s_4\}$ and the lower and upper transition probabilities:

$$L = \begin{pmatrix} 1 & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{6} & \frac{1}{4} & 0 \\ 0 & \frac{1}{4} & \frac{1}{6} & \frac{1}{4} \\ 0 & 0 & \frac{1}{4} & \frac{1}{2} \end{pmatrix}, \tag{43}$$

$$U = \begin{pmatrix} 1 & 0 & 0 & 0 \\ \frac{7}{12} & \frac{5}{12} & \frac{1}{2} & 0 \\ 0 & \frac{7}{12} & \frac{1}{2} & \frac{7}{12} \\ 0 & 0 & \frac{1}{2} & \frac{3}{4} \end{pmatrix}, \tag{44}$$

that is,

$$\boldsymbol{\mathcal{P}}(s_i) = \{\mathbf{P}(s_i) \colon L(s_i) \leq \mathbf{P}(s_i) \leq U(s_i)\} \tag{45}$$

where $L(s_i)$ is the ith row of L, and $U(s_i)$ is the ith row of U. As mentioned, the corresponding transition operators $\underline{\mathrm{T}}$ and $\overline{\mathrm{T}}$ can be efficiently evaluated through linear programming.

We are interested in verifying the property:

$$s_2 \vDash \mathbb{P}_{[0.9,1]}\Big(\Diamond^{\leq 2}\Big(\mathbb{P}_{[0.4,1]}\big((s_2 \vee s_3) \mathbin{\mathsf{U}}^{\leq 6} s_1\big)\Big)\Big) \tag{46}$$

that is, starting from s_2, with probability at least 0.9, in at most two steps we end up in a state from which, with probability at least 0.4, we end up in s_1 in at most 6 steps without visiting s_4.

To answer the above question, we start with evaluating:

$$\mathrm{Sat}(s_2 \vee s_3) = \{s_2, s_3\} \text{ and } \mathrm{Sat}(s_1) = \{s_1\}. \tag{47}$$

Next, we need:

$$\mathrm{Sat}\Big(\mathbb{P}_{[0.4,1]}\big(\{s_2, s_3\} \mathbin{\mathsf{U}}^{\leq 6} \{s_1\}\big)\Big) \tag{48}$$

Clearly, s_1 belongs to the set because:

$$\underline{\Pr}(\{s_2, s_3\} \mathbin{\mathsf{U}}^{\leq 6} \{s_1\} \mid s_1) = \overline{\Pr}(\{s_2, s_3\} \mathbin{\mathsf{U}}^{\leq 6} \{s_1\} \mid s_1) = 1, \tag{49}$$

and s_4 does not belong to the set because:

$$\underline{\Pr}(\{s_2, s_3\} \mathbin{\mathsf{U}}^{\leq 6} \{s_1\} \mid s_4) = \overline{\Pr}(\{s_2, s_3\} \mathbin{\mathsf{U}}^{\leq 6} \{s_1\} \mid s_4) = 0, \tag{50}$$

For s_2,

$$\underline{\Pr}(\{s_2, s_3\} \mathbin{\mathsf{U}}^{\leq 6} \{s_1\} \mid s_2) = \underline{\mathrm{T}}^6_*(s_2, \{s_1\}) \tag{51}$$

$$= 0.4809 \tag{52}$$

$$\overline{\Pr}(\{s_2, s_3\} \mathbin{\mathsf{U}}^{\leq 6} \{s_1\} \mid s_2) = \overline{\mathrm{T}}^6_*(s_2, \{s_1\}) \tag{53}$$

$$= 0.8685 \tag{54}$$

so s_2 belongs to the set, as $[0.4809, 0.8685] \subseteq [0.4, 1]$. For s_3,

$$\underline{\Pr}(\{s_2, s_3\} \mathbin{\mathsf{U}}^{\leq 6} \{s_1\} \mid s_3) = \underline{\mathrm{T}}^6_*(s_2, \{s_1\}) \tag{55}$$

$$= 0.1415 \tag{56}$$

$$\overline{\Pr}(\{s_2, s_3\} \mathbin{\mathsf{U}}^{\leq 6} \{s_1\} \mid s_3) = \overline{\mathrm{T}}^6_*(s_2, \{s_1\}) \tag{57}$$

$$= 0.5934 \tag{58}$$

so s_3 does not belong to the set, as $[0.1415, 0.5934] \not\subseteq [0.4, 1]$. Concluding,

$$\mathrm{Sat}\Big(\mathbb{P}_{[0.4,1]}\big(\{s_2, s_3\} \mathsf{U}^{\leq 6} \{s_1\}\big)\Big) = \{s_1, s_2\}. \quad (59)$$

Whence, finally, we need to calculate

$$\mathrm{Sat}\big(\mathbb{P}_{[0.9,1]}(\mathrm{true}\, \mathsf{U}^{\leq 2} \{s_1, s_2\})\big). \quad (60)$$

In fact, to verify Eq. (46), we only need to determine whether s_2 belongs to this set. Clearly it does, because

$$\underline{\Pr}(\mathrm{true}\, \mathsf{U}^{\leq 2} \{s_1, s_2\} \mid s_2) = \overline{\Pr}(\mathrm{true}\, \mathsf{U}^{\leq 2} \{s_1, s_2\} \mid s_2) = 1, \quad (61)$$

and obviously $\{1\} \subseteq [0.9, 1]$.

5 Conclusion

We have provided a model checking algorithm for imprecise Markov chains that is equally easy as the corresponding algorithm for precise Markov chains. Rather surprisingly, the same bag of tricks from the precise case can be used for the imprecise case.

An interesting open problem is the evaluation of $\Phi \mathsf{U} \Psi$, where there is no bound on number of steps. Intuitively, it seems obvious that we can do this by evaluating $\Phi \mathsf{U}^{\leq n} \Psi$ for large enough n. How large should n be? When is convergence guaranteed? Are there also generally applicable direct methods as in the precise case? We may also need to deal with the technical issue of dealing with a countable number of partitions to express the Markov condition, a case which is not handled by Walley's theory [11, Sec. 8.1], but covered by Williams's approach [13, 14].

Finally, there are additional optimisation tricks possible for precise Markov chains (see for instance [2, Sec. 10.1.1, Remark 10.17]). Even though these are somewhat technical, it would be interesting to see whether we can recycle such tricks for imprecise Markov chains as well.

Acknowledgements

We thank all reviewers for their constructive comments, which have led to some improvements of the paper.

References

[1] Christel Baier, Holger Hermanns, Joost-Pieter Katoen, and Boudewijn R. Haverkort. Efficient computation of time-bounded reachability probabilities in uniform continuous-time Markov decision processes. *Theoretical Computer Science*, 345(1):2–26, November 2005. doi:10.1016/j.tcs.2005.07.022.

[2] Christel Baier and Joost-Pieter Katoen. *Principles of Model Checking*. The MIT Press, 2008.

[3] Sancho E. Berenguer and Robert L. Smith. The expected number of extreme points of a random linear program. *Mathematical Programming*, 35(2):129–134, 1986. doi:10.1007/BF01580643.

[4] George Boole. *An investigation of the laws of thought on which are founded the mathematical theories of logic and probabilities*. Walton and Maberly, London, 1854.

[5] Gert de Cooman, Filip Hermans, and Erik Quaeghebeur. Imprecise Markov chains and their limit behavior. *Probability in the Engineering and Informational Sciences*, 23(4):597–635, October 2009. arXiv:0801.0980, doi:10.1017/S0269964809990039.

[6] D. J. Hartfiel. *Markov Set-Chains*. Springer-Verlag, Berlin, 1998.

[7] D. J. Hartfiel and E. Seneta. On the theory of Markov set-chains. *Advances in Applied Probability*, 26(4):947–964, 1994. doi:10.2307/1427899.

[8] Joost-Pieter Katoen, Daniel Klink, Martin Leucker, and Verena Wolf. Three-valued abstraction for probabilistic systems. *The Journal of Logic and Algebraic Programming*, 81(4):356–389, 2012. doi:10.1016/j.jlap.2012.03.007.

[9] Igor O. Kozine and Lev V. Utkin. Interval-valued finite Markov chains. *Reliable Computing*, 8:97–113, 2002. doi:10.1023/A:1014745904458.

[10] Damjan Škulj. Discrete time Markov chains with interval probabilities. *International Journal of Approximate Reasoning*, 50(8):1314–1329, 2009. doi:10.1016/j.ijar.2009.06.007.

[11] Peter Walley. *Statistical Reasoning with Imprecise Probabilities*. Chapman and Hall, London, 1991.

[12] K. Weichselberger. *Elementare Grundbegriffe einer allgemeineren Wahrscheinlichkeitsrechnung I — Intervallwahrscheinlichkeit als umfassendes Konzept*. Physica, Heidelberg, 2001. In cooperation with Thomas Augustin and Anton Wallner.

[13] Peter M. Williams. Notes on conditional previsions. Technical report, School of Math. and Phys. Sci., Univ. of Sussex, 1975.

[14] Peter M. Williams. Notes on conditional previsions. *International Journal of Approximate Reasoning*, 44(3):366–383, 2007. doi:10.1016/j.ijar.2006.07.019.