Reachability in Parametric Interval Markov Chains Using Constraints

Anicet Bart, Benoit Delahaye, Didier Lime, Eric Monfroy, Charlotte Truchet

Reachability in Parametric Interval Markov Chains using Constraints*

Anicet Bart[1], Benoît Delahaye[2], Didier Lime[3],
Éric Monfroy[2], and Charlotte Truchet[2]

[1] Institut Mines-Télécom Atlantique - LS2N, UMR 6004 - Nantes, France
[2] Université de Nantes - LS2N, UMR 6004 - Nantes, France
[3] École Centrale de Nantes - LS2N, UMR 6004 - Nantes, France
<firstname>.<lastname>@ls2n.fr

Abstract. Parametric Interval Markov Chains (pIMCs) are a specification formalism that extend Markov Chains (MCs) and Interval Markov Chains (IMCs) by taking into account imprecision in the transition probability values: transitions in pIMCs are labeled with parametric intervals of probabilities. In this work, we study the difference between pIMCs and other Markov Chain abstractions models and investigate the two usual semantics for IMCs: once-and-for-all and at-every-step. In particular, we prove that both semantics agree on the maximal/minimal reachability probabilities of a given IMC. We then investigate solutions to several parameter synthesis problems in the context of pIMCs – consistency, qualitative reachability and quantitative reachability – that rely on constraint encodings. Finally, we propose a prototype implementation of our constraint encodings with promising results.

1 Introduction

Discrete time Markov chains (MCs for short) are a standard probabilistic modeling formalism that has been extensively used in the litterature to reason about software [7] and real-life systems [14]. However, when modeling real-life systems, the exact value of transition probabilities may not be known precisely. Several formalisms abstracting MCs have therefore been developed. Parametric Markov chains [1] (pMCs for short) extend MCs by allowing parameters to appear in transition probabilities. In this formalism, parameters are variables and transition probabilities may be expressed as polynomials over these variables. A given pMC therefore represents a potentially infinite set of MCs, obtained by replacing each parameter by a given value. pMCs are particularly useful to represent systems where dependencies between transition probabilities are required. Indeed, a given parameter may appear in several dinstinct transition probabilities, therefore requiring that the same value is given to all its occurences. Interval Markov

* This work is partially supported by the ANR national research programs PACS (ANR-14-CE28-0002) and Coverif (ANR-15-CE25-0002), and the regional programme Atlanstic2020 funded by the French Region Pays de la Loire and the European Regional Development Fund.

chains [15] (IMC for short) extend MCs by allowing precise transition probabilities to be replaced by intervals, but cannot represent dependencies between distinct transitions. IMCs have mainly been studied with two distinct semantics interpretation. Under the *once-and-for-all* semantics, a given IMC represents a potentially infinite number of MCs where transition probabilities are chosen inside the specified intervals while keeping the same underlying graph structure. The *at-every-step* semantics, which was the original semantics given to IMCs in [15], does not require MCs to preserve the underlying graph structure of the original IMC but instead allows an "unfolding" of the original graph structure where different probability values may be chosen (inside the specified interval) at each occurence of the given transition.

Model-checking algorithms and tools have been developed in the context of pMCs [9, 13, 16] and IMCs with the once-and-for-all semantics [12, 5]. State of the art tools [9] for pMC verification compute a rational function on the parameters that characterizes the probability of satisfying a given property, and then use external tools such as SMT solving [9] for computing the satisfying parameter values. For these methods to be viable in practice, the number of parameters used is quite limited. On the other hand, the model-checking procedure for IMCs presented in [5] is adapted from machine learning and builds successive refinements of the original IMCs that optimize the probability of satisfying the given property. This algorithm converges, but not necessarilly to a global optimum. It is worth noticing that existing model checking procedures for pMCs and IMCs strongly rely on their underlying graph structure. As a consequence, to the best of our knowledge, no solutions for model-checking IMCs with the at-every-step semantics have been proposed yet.

In this paper, we focus on Parametric interval Markov chains [11] (pIMCs for short), that generalize both IMCs and pMCs by allowing parameters to appear in the endpoints of the intervals specifying transition probabilities, and we provide four main contributions.

First, we formally compare abstraction formalisms for MCs in terms of succinctness: we show in particular that pIMCs are *strictly more succinct* than both pMCs and pIMCs when equipped with the right semantics. In other words, everything that can be expressed using pMCs or IMCs can also be expressed using pIMCs while the reverse does not hold. Second, we prove that the once-and-for-all and the at-every-step semantics are equivalent w.r.t. rechability properties, both in the IMC and in the pIMC settings. Notably, this result gives theoretical backing to the generalization of existing works on the verification of IMCs to the at-every-step semantics. Third, we study the parametric verification of fundamental properties at the pIMC level: consistency, qualitative reachability, and quantitative reachability. Given the expressivity of the pIMC formalism, the risk of producing a pIMC specification that is incoherent and therefore does not model any concrete MC is high. We therefore propose constraint encodings for deciding whether a given pIMC is consistent and, if so, synthesizing parameter values ensuring consistency. We then extend these encodings to qualitative reachability, *i.e.*, ensuring that given state labels are reachable in *all* (resp. *none*) of

the MCs modeled by a given pIMC. Finally, we focus on the quantitative reachability problem, *i.e.*, synthesizing parameter values such that the probability of reaching given state labels satisfies fixed bounds in *at least one* (resp. *all*) MCs modeled by a given pIMC. While consistency and qualitative reachability for pIMCs have already been studied in [11], the constraint encodings we propose in this paper are significantly smaller (linear instead of exponential). To the best of our knowledge, our results provide the first solution to the quantitative reachability problem for pIMCs. Our last contribution is the implementation of all our verification algorithms in a prototype tool that generates the required constraint encodings and can be plugged to any SMT solver for their resolution. Due to space limitation, proofs and detailed examples are given in [4].

2 Background

In this section we introduce notions and notations that will be used throughout the paper. Given a finite set of variables $X = \{x_1, \ldots, x_k\}$, we write D_x for the domain of the variable $x \in X$ and D_X for the set of domains associated to the variables in X. A valuation v over X is a set $v = \{(x, d) | x \in X, d \in D_x\}$ of elementary valuations (x, d) where for each $x \in X$ there exists a unique pair of the form (x, d) in v. When clear from the context, we write $v(x) = d$ for the value given to variable x according to valuation v. A rational function f over X is a division of two (multivariate) polynomials g_1 and g_2 over X with rational coefficients, *i.e.*, $f = g_1/g_2$. We write $\mathbb{Q}$ the set of rational numbers and $\mathbb{Q}_X$ the set of rational functions over X. The evaluation $v(g)$ of a polynomial g under the valuation v replaces each variable $x \in X$ by its value $v(x)$.

An *atomic constraint* over X is a Boolean expression of the form $f(X) \bowtie g(X)$, with $\bowtie \in \{\leq, \geq, <, >, =\}$ and f and g two functions over variables in X and constants. A constraint is *linear* if the functions f and g are linear. A *constraint* over X is a Boolean combination of atomic constraints over X.

Given a finite set of states S, we write $\mathsf{Dist}(S)$ for the set of probability distributions over S, *i.e.*, the set of functions $\mu : S \to [0, 1]$ such that $\sum_{s \in S} \mu(s) = 1$. We write $\mathbb{I}$ for the set containing all the interval subsets of $[0, 1]$. In the following, we consider a universal set of symbols A that we use for labelling the states of our structures. We call these symbols *atomic propositions*. We will use Latin alphabet in state context and Greek alphabet in atomic proposition context.

Constraints. Constraints are first order logic predicates used to model and solve combinatorial problems [19]. A problem is described with a list of variables, each in a given domain of possible values, together with a list of constraints over these variables. Such problems are then sent to solvers which decide whether the problem is satisfiable, *i.e.*, if there exists a valuation of the variables satisfying all the constraints, and in this case computes a solution. Checking satisfiability of constraint problems is difficult in general, as the space of all possible valuations has a size exponential in the number of variables.

Formally, a Constraint Satisfaction Problem (CSP) is a tuple $\Omega = (X, D, C)$ where X is a finite set of variables, $D = D_X$ is the set of all the domains

associated to the variables from X, and C is a set of constraints over X. We say that a valuation over X satisfies Ω if and only if it satisfies all the constraints in C. We write $v(C)$ for the satisfaction result of the valuation of the constraints C according to v (*i.e.*, true or false). In the following we call CSP *encoding* a scheme for formulating a given problem into a CSP. The size of a CSP corresponds to the number of variables and atomic constraints appearing in the problem. Note that, in constraint programming, having less variables or less constraints during the encoding does not necessarily imply faster solving time of the problems.

Discrete Time Markov Chains. A Discrete Time Markov Chain (DTMC or MC for short) is a tuple $\mathcal{M} = (S,\ s_0,\ p,\ V)$, where S is a finite set of states containing the initial state s_0, $V : S \to 2^A$ is a labelling function, and $p : S \to \mathsf{Dist}(S)$ is a probabilistic transition function. We write $\mathtt{MC}$ for the set containing all the discrete time Markov chains.

A Markov Chain can be seen as a directed graph where the nodes correspond to the states of the MC and the edges are labelled with the probabilities given by the transition function of the MC. In this representation, a missing transition between two states represents a transition probability of zero. As usual, given a MC $\mathcal{M}$, we call a *path* of $\mathcal{M}$ a sequence of states obtained from executing $\mathcal{M}$, *i.e.*, a sequence $\omega = s_1, s_2, \ldots$ s.t. the probability of taking the transition from s_i to s_{i+1} is strictly positive, $p(s_i)(s_i + 1) > 0$, for all i. A path ω is finite iff it belongs to S^*, *i.e.*, it represents a finite sequence of transitions from $\mathcal{M}$.

Example 1. Figure 1 illustrates the Markov chain $\mathcal{M}_1 = (S, s_0, p, V) \in \mathtt{MC}$ where $S = \{s_0, s_1, s_2, s_3, s_4\}$, the atomic proposition are restricted to $\{\alpha, \beta\}$, the initial state is s_0, and the labelling function V corresponds to $\{(s_0, \emptyset), (s_1, \alpha), (s_2, \beta), (s_3, \{\alpha, \beta\}), (s_4, \alpha)\}$. The sequences of states (s_0, s_1, s_2), (s_0, s_2), and (s_0, s_2, s_2, s_2), are three (finite) paths from the initial state s_0 to the state s_2.

Reachability. A Markov chain $\mathcal{M}$ defines a unique probability measure $\mathbb{P}^{\mathcal{M}}$ over the paths from $\mathcal{M}$. According to this measure, the probability of a finite path $\omega = s_0, s_1, \ldots, s_n$ in $\mathcal{M}$ is the product of the probabilities of the transitions executed along this path, *i.e.*, $\mathbb{P}^{\mathcal{M}}(\omega) = p(s_0)(s_1) \cdot p(s_1)(s_2) \cdot \ldots \cdot p(s_{n-1})(s_n)$. This distribution naturally extends to infinite paths (see [2]) and to sequences of states over S that are not paths of $\mathcal{M}$ by giving them a zero probability.

Given a MC $\mathcal{M}$, the overall probability of reaching a given state s from the initial state s_0 is called the *reachability probability* and written $\mathbb{P}^{\mathcal{M}}_{s_0}(\Diamond s)$ or $\mathbb{P}^{\mathcal{M}}(\Diamond s)$ when clear from the context. This probability is computed as the sum of the probabilities of all finite paths starting in the initial state and reaching this state for the first time. Formally, let $\mathsf{reach}_{s_0}(s) = \{\omega \in S^* \mid \omega = s_0, \ldots s_n$ with $s_n = s$ and $s_i \neq s\ \forall 0 \leq i < n\}$ be the set of such paths. We then define $\mathbb{P}^{\mathcal{M}}(\Diamond s) = \sum_{\omega \in \mathsf{reach}_{s_0}(s)} \mathbb{P}^{\mathcal{M}}(\omega)$ if $s \neq s_0$ and 1 otherwise. This notation naturally extends to the reachability probability of a state s from a state t that is not s_0, written $\mathbb{P}^{\mathcal{M}}_t(\Diamond s)$ and to the probability of reaching a label $\alpha \subseteq A$ written $\mathbb{P}^{\mathcal{M}}_{s_0}(\Diamond \alpha)$. In the following, we say that a state s (resp. a label $\alpha \subseteq A$) is reachable in $\mathcal{M}$ iff the reachability probability of this state (resp. label) from the initial state is strictly positive.

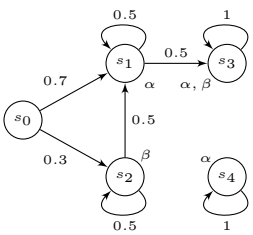

Fig. 1: MC $\mathcal{M}_1$

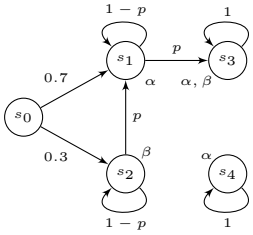

Fig. 2: pMC $\mathcal{I}'$

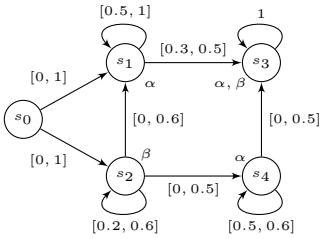

Fig. 3: IMC $\mathcal{I}$

Example 2 (Example 1 continued). In Figure 1 the probability of the path (s_0, s_2, s_1, s_1, s_3) is $0.3 \cdot 0.5 \cdot 0.5 \cdot 0.5 = 0.0375$ and the probability of reaching the state s_1 is $\mathbb{P}^{\mathcal{M}_1}(\Diamond s_1) = p(s_0)(s_1) + \Sigma_{i=0}^{+\infty} p(s_0)(s_2) \cdot p(s_2)(s_2)^i \cdot p(s_2)(s_1) = p(s_0)(s_1) + p(s_0)(s_2) \cdot p(s_2)(s_1) \cdot (1/(1 - p(s_2)(s_2))) = 1$. Furthermore, the probability of reaching β corresponds to the probability of reaching the state s_2.

3 Markov Chains Abstractions

Modelling an application as a Markov Chain requires knowing the exact probability for each possible transition of the system. However, this can be difficult to compute or to measure in the case of a real-life application (*e.g.*, precision errors, limited knowledge). In this section, we start with a generic definition of Markov chain abstraction models. Then we recall three abstraction models from the literature, respectively pMC, IMC, and pIMC, and finally we present a comparison of these existing models in terms of succinctness.

Definition 1 (Markov chain Abstraction Model). *A Markov chain abstraction model (an abstraction model for short) is a pair* $(\mathtt{L}, \models)$ *where* $\mathtt{L}$ *is a nonempty set and* $\models$ *is a relation between* $\mathtt{MC}$ *and* $\mathtt{L}$*. Let* $\mathcal{P}$ *be in* $\mathtt{L}$ *and* $\mathcal{M}$ *be in* $\mathtt{MC}$ *we say that* $\mathcal{M}$ *implements* $\mathcal{P}$ *iff* $(\mathcal{M}, \mathcal{P})$ *belongs to* $\models$ *(*i.e.*,* $\mathcal{M} \models \mathcal{P}$*). When the context is clear, we do not mention the satisfaction relation* $\models$ *and only use* $\mathtt{L}$ *to refer to the abstraction model* $(\mathtt{L}, \models)$*.*

A *Markov chain Abstraction Model* is a specification theory for MCs. It consists in a set of abstract objects, called *specifications*, each of which representing a (potentially infinite) set of MCs – *implementations* – together with a satisfaction relation defining the link between implementations and specifications. As an example, consider the powerset of $\mathtt{MC}$ (*i.e.*, the set containing all the possible sets of Markov chains). Clearly, $(2^{\mathtt{MC}}, \in)$ is a Markov chain abstraction model, which we call the *canonical abstraction model*. This abstraction model has the advantage of representing all the possible sets of Markov chains but it also has the disadvantage that some Markov chain abstractions are only representable by an infinite extension representation. Indeed, recall that there exists subsets of $[0, 1] \subseteq \mathbb{R}$ which cannot be represented in a finite space (e.g., the Cantor set [6]). We now present existing MC abstraction models from the literature.

3.1 Existing MC Abstraction Models

Parametric Markov Chain is a MC abstraction model from [1] where a transition can be annotated by a rational function over *parameters*. We write pMC for the set containing all the parametric Markov chains.

Definition 2 (Parametric Markov Chain). *A Parametric Markov Chain (*pMC *for short) is a tuple $\mathcal{I} = (S, s_0, P, V, Y)$ where S, s_0, and V are defined as for* MC*s, Y is a set of variables (parameters), and $P : S \times S \to \mathbb{Q}_Y$ associates with each potential transition a parameterized probability.*

Let $\mathcal{M} = (S, s_0, p, V)$ be a MC and $\mathcal{I} = (S, s_0, P, V, Y)$ be a pMC. The satisfaction relation $\models_{\mathtt{p}}$ between MC and pMC is defined by $\mathcal{M} \models_{\mathtt{p}} \mathcal{I}$ iff there exists a valuation v of Y s.t. $p(s)(s')$ equals $v(P(s, s'))$ for all s, s' in S.

Example 3. Figure 2 shows a pMC $\mathcal{I}' = (S, s_0, P, V, Y)$ where S, s_0, and V are similar to the same entities in the MC $\mathcal{M}$ from Figure 1, the set of variable Y contains only one variable p, and the parametric transitions in P are given by the edge labelling (*e.g.*, $P(s_0, s_1) = 0.7$, $P(s_1, s_3) = p$, and $P(s_2, s_2) = 1 - p$). Note that the pMC $\mathcal{I}'$ is a specification containing the MC $\mathcal{M}$ from Figure 1.

Interval Markov Chains extend MCs by allowing to label transitions with intervals of possible probabilities instead of precise probabilities. We write IMC for the set containing all the interval Markov chains.

Definition 3 (Interval Markov Chain [15]). *An Interval Markov Chain (*IMC *for short) is a tuple $\mathcal{I} = (S, s_0, P, V)$, where S, s_0, and V are defined as for* MC*s, and $P : S \times S \to \mathbb{I}$ associates with each potential transition an interval of probabilities.*

Example 4. Figure 3 illustrates IMC $\mathcal{I} = (S, s_0, P, V)$ where S, s_0, and V are similar to the MC given in Figure 1. By observing the edge labelling we see that $P(s_0, s_1) = [0, 1]$, $P(s_1, s_1) = [0.5, 1]$, and $P(s_3, s_3) = [1, 1]$. On the other hand, the intervals of probability for missing transitions are reduced to $[0, 0]$, e.g., $P(s_0, s_0) = [0, 0]$, $P(s_0, s_3) = [0, 0]$, $P(s_1, s_4) = [0, 0]$.

In the literature, IMCs have been mainly used with two distinct semantics: *at-every-step* and *once-and-for-all*. Both semantics are associated with distinct satisfaction relations which we now introduce.

The *once-and-for-all* IMC semantics ([9, 20, 18]) is alike to the semantics for pMC, as introduced above. The associated satisfaction relation $\models^o_{\mathtt{I}}$ is defined as follows: A MC $\mathcal{M} = (T, t_0, p, V^M)$ satisfies an IMC $\mathcal{I} = (S, s_0, P, V^I)$ iff $(T, t_0, V^M) = (S, s_0, V^I)$ and for all reachable state s and all state $s' \in S$, $p(s)(s') \in P(s, s')$. In this sense, we say that MC implementations using the once-and-for-all semantics need to have the same structure as the IMC specification.

On the other hand, the *at-every-step* IMC semantics, first introduced in [15], operates as a simulation relation based on the transition probabilities and state labels, and therefore allows MC implementations to have a different structure

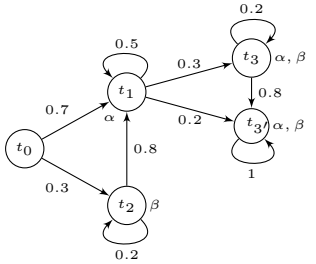

Fig. 4: MC $\mathcal{M}_2$ satisfying the IMC $\mathcal{I}$ from Figure 3 with a different structure

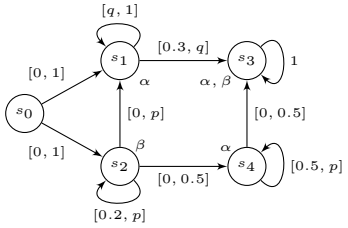

Fig. 5: pIMC $\mathcal{P}$

than the IMC specification. The associated satisfaction relation $\models^a_{\mathtt{I}}$ is defined as follows: A MC $\mathcal{M} = (T, t_0,\ p,\ V^M)$ satisfies an IMC $\mathcal{I} = (S, s_0, P, V^I)$ iff there exists a relation $\mathcal{R} \subseteq T \times S$ such that $(t_0, s) \in \mathcal{R}$ and whenever $(t, s) \in \mathcal{R}$, we have 1. the labels of s and t correspond: $V^M(t) = V^I(s)$, 2. there exists a correspondence function $\delta : T \to (S \to [0, 1])$ s.t. (a) $\forall t' \in T$ if $p(t)(t') > 0$ then $\delta(t')$ is a distribution on S (b) $\forall s' \in S : (\Sigma_{t' \in T} p(t)(t') \cdot \delta(t')(s')) \in P(s, s')$, and (c) $\forall (t', s') \in T \times S$, if $\delta(t')(s') > 0$, then $(t', s') \in \mathcal{R}$. By construction, it is clear that $\models^a_{\mathtt{I}}$ is more general than $\models^o_{\mathtt{I}}$, *i.e.*, that whenever $\mathcal{M} \models^o_{\mathtt{I}} \mathcal{I}$, we also have $\mathcal{M} \models^a_{\mathtt{I}} \mathcal{I}$. The reverse is obviously not true in general, even when the underlying graphs of $\mathcal{M}$ and $\mathcal{I}$ are isomorphic (see [4] for details).

Example 5 (Example 4 continued). Consider the MC $\mathcal{M}_1$ with state space S from Figure 1 and the MC $\mathcal{M}_2$ with state space T from Figure 4. They both satisfy the IMC $\mathcal{I}$ with state space S given in Figure 3. Furthermore, $\mathcal{M}_1$ satisfies $\mathcal{I}$ with the same structure. On the other hand, for the MC $\mathcal{M}_2$ given in Figure 4, the state s_3 from $\mathcal{I}$ has been "split" into two states t_3 and $t_{3'}$ in $\mathcal{M}_2$ and the state t_1 from $\mathcal{M}_2$ "aggregates" states s_1 and s_4 in $\mathcal{I}$. The relation $\mathcal{R} \subseteq T \times S$ containing the pairs (t_0, s_0), (t_1, s_1), (t_1, s_4), (t_2, s_2), (t_3, s_3), and $(t_{3'}, s_3)$ is a satisfaction relation between $\mathcal{M}_2$ and $\mathcal{I}$.

Parametric Interval Markov Chains, as introduced in [11], abstract IMCs by allowing (combinations of) parameters to be used as interval endpoints in IMCs. Under a given parameter valuation the pIMC yields an IMC as introduced above. pIMCs therefore allow the representation, in a compact way and with a finite structure, of a potentially infinite number of IMCs. Note that one parameter can appear in several transitions at once, requiring the associated transition probabilities to depend on one another. Let Y be a finite set of parameters and v be a valuation over Y. By combining notations used for IMCs and pMCs the set $\mathbb{I}(\mathbb{Q}_Y)$ contains all parametrized intervals over $[0, 1]$, and for all $I = [f_1, f_2] \in \mathbb{I}(\mathbb{Q}_Y)$, $v(I)$ denotes the interval $[v(f_1), v(f_2)]$ if $0 \leq v(f_1) \leq v(f_2) \leq 1$ and the empty set otherwise[4]. We write `pIMC` for the set containing all the parametric interval Markov chains.

[4] Indeed, when $0 \leq v(f_1) \leq v(f_2) \leq 1$ is not respected, the interval is inconsistent and therefore empty.

Definition 4 (Parametric Interval Markov Chain [11]). *A Parametric Interval Markov Chain (*pIMC *for short) is a tuple* $\mathcal{P} = (S, s_0, P, V, Y)$*, where* S*,* s_0*,* V *and* Y *are defined as for* pMC*s, and* $P : S \times S \to \mathbb{I}(\mathbb{Q}_Y)$ *associates with each potential transition a (parametric) interval.*

In [11] the authors introduced pIMCs where parametric interval endpoints are limited to linear combination of parameters. In this paper we extend the pIMC model by allowing rational functions over parameters as endpoints of parametric intervals. Given a pIMC $\mathcal{P} = (S, s_0, P, V, Y)$ and a valuation v, we write $v(\mathcal{P})$ for the IMC (S, s_0, P_v, V) obtained by replacing the transition function P from $\mathcal{P}$ with the function $P_v : S \times S \to \mathbb{I}$ defined by $P_v(s, s') = v(P(s, s'))$ for all $s, s' \in S$. The IMC $v(\mathcal{P})$ is called an *instance* of pIMC $\mathcal{P}$. Finally, depending on the semantics chosen for IMCs, two satisfaction relations can be defined between MCs and pIMCs. They are written $\models^a_{\mathtt{pI}}$ and $\models^o_{\mathtt{pI}}$ and defined as follows: $\mathcal{M} \models^a_{\mathtt{pI}} \mathcal{P}$ (resp. $\models^o_{\mathtt{pI}}$) iff there exists an IMC $\mathcal{I}$ instance of $\mathcal{P}$ s.t. $\mathcal{M} \models^a_{\mathtt{I}} \mathcal{I}$ (resp. $\models^o_{\mathtt{I}}$).

Example 6. Consider the pIMC $\mathcal{P} = (S,0, P, V, Y)$ given in Figure 5. The set of states S and the labelling function are the same as in the MC and the IMC presented in Figures 1 and 3 respectively. The set of parameters Y has two elements p and q. Finally, the parametric intervals from the transition function P are given by the edge labelling (*e.g.*, $P(s_1, s_3) = [0.3, q]$, $P(s_2, s_4) = [0, 0.5]$, and $P(s_3, s_3) = [1, 1]$). Note that the IMC $\mathcal{I}$ from Figure 3 is an instance of $\mathcal{P}$ (by assigning the value 0.6 to the parameter p and 0.5 to q). Furthermore, as said in Example 5, the Markov Chains $\mathcal{M}_1$ and $\mathcal{M}_2$ (from Figures 1 and 4 respectively) satisfy $\mathcal{I}$, therefore $\mathcal{M}_1$ and $\mathcal{M}_2$ satisfy $\mathcal{P}$.

In the following, we consider that the size of a pMC, IMC, or pIMC corresponds to its number of states plus its number of transitions not reduced to 0, $[0, 0]$ or $\emptyset$. We will also often need to consider the predecessors (`Pred`), and the successors (`Succ`) of some given states. Given a pIMC with a set of states S, a state s in S, and a subset S' of S, we write:

- $\mathtt{Pred}(s) = \{s' \in S \mid P(s', s) \notin \{\emptyset, [0, 0]\}\}$
- $\mathtt{Succ}(s) = \{s' \in S \mid P(s, s') \notin \{\emptyset, [0, 0]\}\}$
- $\mathtt{Pred}(S') = \bigcup_{s' \in S'} \mathtt{Pred}(s')$
- $\mathtt{Succ}(S') = \bigcup_{s' \in S'} \mathtt{Succ}(s')$

3.2 Abstraction Model Comparisons

IMC, pMC, and pIMC are three Markov chain Abstraction Models. In order to compare their expressiveness and compactness, we introduce the comparison operators $\sqsubseteq$ and $\equiv$. Let $(\mathtt{L}_1, \models_1)$ and $(\mathtt{L}_2, \models_2)$ be two Markov chain abstraction models containing respectively $\mathcal{L}_1$ and $\mathcal{L}_2$. We say that $\mathcal{L}_1$ is entailed by $\mathcal{L}_2$, written $\mathcal{L}_1 \sqsubseteq \mathcal{L}_2$, iff all the MCs satisfying $\mathcal{L}_1$ satisfy $\mathcal{L}_2$ modulo bisimilarity. (*i.e.*, $\forall \mathcal{M} \models_1 \mathcal{L}_1, \exists \mathcal{M}' \models_2 \mathcal{L}_2$ s.t. $\mathcal{M}$ is bisimilar to $\mathcal{M}'$). We say that $\mathcal{L}_1$ is (semantically) equivalent to $\mathcal{L}_2$, written $\mathcal{L}_1 \equiv \mathcal{L}_2$, iff $\mathcal{L}_1 \sqsubseteq \mathcal{L}_2$ and $\mathcal{L}_2 \sqsubseteq \mathcal{L}_1$. Definition 5 introduces succinctness based on the sizes of the abstractions.

Definition 5 (Succinctness). *Let* $(\mathtt{L}_1, \models_1)$ *and* $(\mathtt{L}_2, \models_2)$ *be two Markov chain abstraction models.* $\mathtt{L}_1$ *is at least as succinct as* $\mathtt{L}_2$*, written* $\mathtt{L}_1 \leq \mathtt{L}_2$*, iff there*

exists a polynomial p such that for every $\mathcal{L}_2 \in \mathtt{L}_2$, there exists $\mathcal{L}_1 \in \mathtt{L}_1$ s.t. $\mathcal{L}_1 \equiv \mathcal{L}_2$ and $|\mathcal{L}_1| \leq p(|\mathcal{L}_2|)$.[5] *Thus, $\mathtt{L}_1$ is strictly more succinct than $\mathtt{L}_2$, written $\mathtt{L}_1 < \mathtt{L}_2$, iff $\mathtt{L}_1 \leq \mathtt{L}_2$ and $\mathtt{L}_2 \not\leq \mathtt{L}_1$.*

We start with a comparison of the succinctness of the pMC and IMC abstractions. Since pMCs allow the expression of dependencies between the probabilities assigned to distinct transitions while IMCs allow all transitions to be independant, it is clear that there are pMCs without any equivalent IMCs (regardless of the IMC semantics used), therefore $(\mathtt{IMC}, \models^{\mathtt{o}}_{\mathtt{I}}) \not\leq \mathtt{pMC}$ and $(\mathtt{IMC}, \models^{\mathtt{a}}_{\mathtt{I}}) \not\leq \mathtt{pMC}$ (see [4] for details). On the other hand, IMCs imply that transition probabilities need to satisfy linear inequalities in order to fit given intervals. However, these types of constraints are not allowed in pMCs. It is therefore easy to exhibit IMCs that, regardless of the semantics considered, do not have any equivalent pMC specification. As a consequence, $\mathtt{pMC} \not\leq (\mathtt{IMC}, \models^{\mathtt{o}}_{\mathtt{I}})$ and $\mathtt{pMC} \not\leq (\mathtt{IMC}, \models^{\mathtt{a}}_{\mathtt{I}})$.

We now compare pMCs and IMCs to pIMCs. Recall that the pIMC model is a Markov chain abstraction model allowing to declare parametric interval transitions, while the pMC model allows only parametric transitions (without intervals), and the IMC model allows interval transitions without parameters. Clearly, any pMC and any IMC can be translated into a pIMC with the right semantics (once-and-for-all for pMCs and the chosen IMC semantics for IMCs). This means that $(\mathtt{pIMC}, \models^{o}_{\mathtt{pI}})$ is more succinct than pMC and pIMC is more succinct than IMC for both semantics. Furthermore, since pMC and IMC are not comparable due to the above results, we have that the pIMC abstraction model is strictly more succinct than the pMC abstraction model and than the IMC abstraction model with the right semantics. Our comparison results are presented in Proposition 1. Further explanations and examples are given in [4].

Proposition 1. *The Markov chain abstraction models can be ordered as follows w.r.t. succinctness:* $(\mathtt{pIMC}, \models^{o}_{\mathtt{pI}}) < (\mathtt{pMC}, \models_{\mathtt{p}})$, $(\mathtt{pIMC}, \models^{o}_{\mathtt{pI}}) < (\mathtt{IMC}, \models^{\mathtt{o}}_{\mathtt{I}})$ *and* $(\mathtt{pIMC}, \models^{a}_{\mathtt{pI}}) < (\mathtt{IMC}, \models^{\mathtt{a}}_{\mathtt{I}})$.

Note that $(\mathtt{pMC}, \models_{\mathtt{p}}) \leq (\mathtt{IMC}, \models^{\mathtt{o}}_{\mathtt{I}})$ could be achieved by adding unary constraints on the parameters of a pMC, which is not allowed here. However, this would not have any impact on our other results.

4 Qualitative Properties

As seen above, pIMCs are a succinct abstraction formalism for MCs. The aim of this section is to investigate qualitative properties for pIMCs, *i.e.*, properties that can be evaluated at the specification (pIMC) level, but that entail properties on its MC implementations. pIMC specifications are very expressive as they allow the abstraction of transition probabilities using both intervals and parameters. Unfortunately, as it is the case for IMCs, this allows the expression of incorrect specifications. In the IMC setting, this is the case either when some

[5] $|\mathcal{L}_1|$ and $|\mathcal{L}_2|$ are the sizes of $\mathcal{L}_1$ and $\mathcal{L}_2$, respectively.

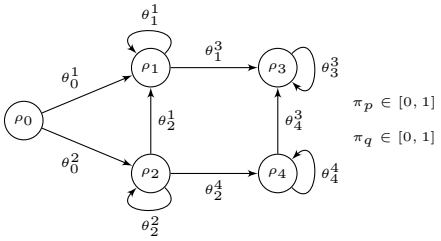

Fig. 6: Variables in the CSP produced by $\mathbf{C}_{\exists\mathtt{c}}$ for the pIMC $\mathcal{P}$ from Fig. 5

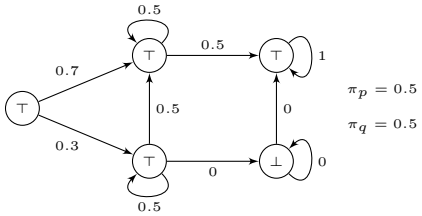

Fig. 7: A solution to the CSP $\mathbf{C}_{\exists\mathtt{c}}(\mathcal{P})$ for the pIMC $\mathcal{P}$ from Fig. 5

intervals are ill-formed or when there is no probability distribution matching the interval constraints of the outgoing transitions of some reachable state. In this case, no MC implementation exists that satisfies the IMC specification. Deciding whether an implementation that satisfies a given specification exists is called the consistency problem. In the pIMC setting, the consistency problem is made more complex because of the parameters which can also induce inconsistencies in some cases. One could also be interested in verifying whether there exists an implementation that reaches some target states/labels, and if so, propose a parameter valuation ensuring this property. Both the consistency and the consistent reachability problems have already been investigated in the IMC and pIMC setting [10, 11]. In this section, we briefly recall these problems and propose new solutions based on CSP encodings. Our encodings are linear in the size of the original pIMCs whereas the algorithms from [10, 11] are exponential.

4.1 Existential Consistency

A pIMC $\mathcal{P}$ is existential consistent iff there exists a MC $\mathcal{M}$ satisfying $\mathcal{P}$ (*i.e.*, there exists a MC $\mathcal{M}$ satisfying an IMC $\mathcal{I}$ instance of $\mathcal{P}$). As seen in Section 2, pIMCs are equipped with two semantics: once-and-for-all ($\models^o_{\mathtt{pI}}$) and at-every-step ($\models^a_{\mathtt{pI}}$). Recall that $\models^o_{\mathtt{pI}}$ imposes that the underlying graph structure of implementations needs to be isomorphic to the graph structure of the corresponding specification. In contrast, $\models^a_{\mathtt{pI}}$ allows implementations to have a different graph structure. It therefore seems that some pIMCs could be inconsistent w.r.t $\models^o_{\mathtt{pI}}$ while being consistent w.r.t $\models^a_{\mathtt{pI}}$. On the other hand, checking the consistency w.r.t $\models^o_{\mathtt{pI}}$ seems easier because of the fixed graph structure.

In [10], the author firstly proved that both semantics are equivalent w.r.t. existential consistency, and proposed a CSP encoding for verifying this property which is exponential in the size of the pIMC. Based on this result of semantics equivalence w.r.t. existential consistency from [10] we propose a new CSP encoding, written $\mathbf{C}_{\exists\mathtt{c}}$, for verifying the existential consistency property for pIMCs.

Let $\mathcal{P} = (S,s_0,P,V,Y)$ be a pIMC, we write $\mathbf{C}_{\exists\mathtt{c}}(\mathcal{P})$ for the CSP produced by $\mathbf{C}_{\exists\mathtt{c}}$ according to $\mathcal{P}$. Any solution of $\mathbf{C}_{\exists\mathtt{c}}(\mathcal{P})$ will correspond to a MC satisfying $\mathcal{P}$. In $\mathbf{C}_{\exists\mathtt{c}}(\mathcal{P})$, we use one variable π_p with domain $[0,1]$ per parameter p in Y; one variable $\theta_s^{s'}$ with domain $[0,1]$ per transition (s,s') in $\{\{s\}\times\mathtt{Succ}(s) \mid s \in S\}$; and

one Boolean variable ρ_s per state s in S. These Boolean variables will indicate for each state whether it appears in the MC solution of the CSP (*i.e.*, in the MC satisfying the pIMC $\mathcal{P}$). For each state $s \in S$, Constraints are as follows:

(1) ρ_s, if $s = s_0$
(2) $\rho_s \Leftrightarrow \Sigma_{s' \in \mathtt{Succ}(s)} \theta_s^{s'} = 1$
(3) $\neg\rho_s \Leftrightarrow \Sigma_{s' \in \mathtt{Pred}(s) \setminus \{s\}} \theta_{s'}^{s} = 0$, if $s \neq s_0$
(4) $\neg\rho_s \Leftrightarrow \Sigma_{s' \in \mathtt{Succ}(s)} \theta_s^{s'} = 0$
(5) $\rho_s \Rightarrow \theta_s^{s'} \in P(s, s')$, for all $s' \in \mathtt{Succ}(s)$

Recall that given a pIMC $\mathcal{P}$ the objective of the CSP $\mathbf{C}_{\exists\mathbf{c}}(\mathcal{P})$ is to construct a MC $\mathcal{M}$ satisfying $\mathcal{P}$. Constraint **(1)** states that the initial state s_0 appears in $\mathcal{M}$. Constraint **(3)** ensures that for each non-initial state s, variable ρ_s is set to false iff s is not reachable from its predecessors. Constraint **(2)** ensures that if a state s appears in $\mathcal{M}$, then its outgoing transitions form a probability distribution. On the contrary, Constraint **(4)** propagates non-appearing states (*i.e.*, if a state s does not appear in $\mathcal{M}$ then all its outgoing transitions are set to zero). Finally, Constraint **(5)** states that, for all appearing states, the outgoing transition probabilities must be selected inside the specified intervals.

Example 7. Consider the pIMC $\mathcal{P}$ given in Figure 5. Figure 6 describes the variables in $\mathbf{C}_{\exists\mathbf{c}}(\mathcal{P})$: one variable per transition (*e.g.*, θ_0^1, θ_0^2, θ_1^1), one Boolean variable per state (*e.g.*, ρ_0, ρ_1), and one variable per parameter (π_p and π_q). The following constraints correspond to the Constraints **(2)**, **(3)**, **(4)**, and **(5)** generated by our encoding $\mathbf{C}_{\exists\mathbf{c}}$ for the state 2 of $\mathcal{P}$:

$\neg\rho_2 \Leftrightarrow \theta_0^2 = 0$ $\quad$ $\rho_2 \Leftrightarrow \theta_2^1 + \theta_2^2 + \theta_2^4 = 1$ $\quad$ $\rho_2 \Rightarrow 0.2 \leq \theta_2^2 \leq \pi_p$
$\neg\rho_2 \Leftrightarrow \theta_2^1 + \theta_2^2 + \theta_2^4 = 0$ $\quad$ $\rho_2 \Rightarrow 0 \leq \theta_2^1 \leq \pi_p$ $\quad$ $\rho_2 \Rightarrow 0 \leq \theta_2^4 \leq 0.5$

Finally, Figure 7 describes a solution for the CSP $\mathbf{C}_{\exists\mathbf{c}}(\mathcal{P})$. Note that given a solution of a pIMC encoded by $\mathbf{C}_{\exists\mathbf{c}}$, one can construct a MC satisfying the given pIMC by keeping all the states s s.t. ρ_s is equal to true and considering the transition function given by the probabilities in the $\theta_s^{s'}$ variables. We now show that our encoding works as expected.

Proposition 2. *A* pIMC $\mathcal{P}$ *is existential consistent iff* $\mathbf{C}_{\exists\mathbf{c}}(\mathcal{P})$ *is satisfiable.*

Our existential consistency encoding is linear in the size of the pIMC instead of exponential for the encoding from [11] which enumerates the powerset of the states in the pIMC resulting in deep nesting of conjunctions and disjunctions.

4.2 Qualitative Reachability

Let $\mathcal{P} = (S, s_0, P, V, Y)$ be a pIMC and $\alpha \subseteq A$ be a state label. We say that α is *existential reachable* in $\mathcal{P}$ iff there exists an implementation $\mathcal{M}$ of $\mathcal{P}$ where α is reachable (*i.e.*, $\mathbb{P}^{\mathcal{M}}(\Diamond\alpha) > 0$). In a dual way, we say that α is *universal reachable* in $\mathcal{P}$ iff α is reachable in any implementation $\mathcal{M}$ of $\mathcal{P}$. As for existential consistency, we use a result from [10] that states that both pIMC semantics are equivalent w.r.t. existential (and universal) reachability. We therefore propose a new CSP encoding, written $\mathbf{C}_{\exists\mathbf{r}}$, that extends $\mathbf{C}_{\exists\mathbf{c}}$, for verifying these properties. Formally, CSP $\mathbf{C}_{\exists\mathbf{r}}(\mathcal{P}) = (X \cup X', D \cup D', C \cup$

C') is such that $(X, D, C) = \mathbf{C}_{\exists\mathbf{c}}(\mathcal{P})$, X' contains one integer variable ω_s with domain $[0, |S|]$ per state s in S, D' contains the domains of these variables, and C' is composed of the following constraints for each state $s \in S$:

(6) $\omega_s = 1$, if $s = s_0$ **(7)** $\omega_s \neq 1$, if $s \neq s_0$ **(8)** $\rho_s \Leftrightarrow (\omega_s \neq 0)$

(9) $\omega_s > 1 \Rightarrow \bigvee_{s' \in \texttt{Pred}(s) \setminus \{s\}} (\omega_s = \omega_{s'} + 1) \wedge (\theta_s^{s'} > 0)$, if $s \neq s_0$

(10) $\omega_s = 0 \Leftrightarrow \bigwedge_{s' \in \texttt{Pred}(s) \setminus \{s\}} (\omega_{s'} = 0) \vee (\theta_s^{s'} = 0)$, if $s \neq s_0$

Recall first that CSP $\mathbf{C}_{\exists\mathbf{c}}(P)$ constructs a Markov chain $\mathcal{M}$ satisfying $\mathcal{P}$. Informally, for each state s in $\mathcal{M}$ the Constraints **(6)**, **(7)**, **(9)** and **(10)** in $\mathbf{C}_{\exists\mathbf{r}}$ ensure that $\omega_s = k$ iff there exists in $\mathcal{M}$ a path from the initial state to s of length $k-1$ with non zero probability; and state s is not reachable in $\mathcal{M}$ from the initial state s_0 iff ω_s equals to 0. Finally, Constraint **(8)** enforces the Boolean reachability indicator variable ρ_s to bet set to true iff there exists a path with non zero probability in $\mathcal{M}$ from the initial state s_0 to s (*i.e.*, $\omega_s \neq 0$).

Let S_α be the set of states from $\mathcal{P}$ labeled with α. $\mathbf{C}_{\exists\mathbf{r}}(\mathcal{P})$ therefore produces a Markov chain satisfying $\mathcal{P}$ where reachable states s are such that $\rho_s = \mathsf{true}$. As a consequence, α is existential reachable in $\mathcal{P}$ iff $\mathbf{C}_{\exists\mathbf{r}}(\mathcal{P})$ admits a solution such that $\bigvee_{s \in S_\alpha} \rho_s$; and α is universal reachable in $\mathcal{P}$ iff $\mathbf{C}_{\exists\mathbf{r}}(\mathcal{P})$ admits no solution such that $\bigwedge_{s \in S_\alpha} \neg\rho_s$. This is formalised in the following proposition.

Proposition 3. *Let $\mathcal{P} = (S, s_0, P, V, Y)$ be a* pIMC*, $\alpha \subseteq A$ be a state label, $S_\alpha = \{s \mid V(s) = \alpha\}$, and (X, D, C) be the* CSP *$\mathbf{C}_{\exists\mathbf{r}}(\mathcal{P})$.*

- CSP $(X, D, C \cup \bigvee_{s \in S_\alpha} \rho_s)$ *is satisfiable iff α is existential reachable in $\mathcal{P}$*
- CSP $(X, D, C \cup \bigwedge_{s \in S_\alpha} \neg\rho_s)$ *is unsatisfiable iff α is universal reachable in $\mathcal{P}$*

As for the existential consistency problem, we have an exponential gain in terms of size of the encoding compared to [11]: the number of constraints and variables in $\mathbf{C}_{\exists\mathbf{r}}$ is linear in terms of the size of the encoded pIMC.

Remark. In $\mathbf{C}_{\exists\mathbf{r}}$ Constraints **(3)** inherited from $\mathbf{C}_{\exists\mathbf{c}}$ are entailed by Constraints **(8)** and **(10)** added to $\mathbf{C}_{\exists\mathbf{r}}$. Thus, in a practical approach one may ignore Constraints **(3)** from $\mathbf{C}_{\exists\mathbf{c}}$ if they do not improve the solver performances.

5 Quantitative Properties

We now move to the verification of quantitative reachability properties in pIMCs. Quantitative reachability has already been investigated in the context of pMCs and IMCs with the once-and-for-all semantics. Due to the complexity of allowing implementation structures to differ from the structure of the specifications, quantitative reachability in IMCs with the at-every-step semantics has, to the best of our knowledge, never been studied. In this section, we propose our main theoretical contribution: a theorem showing that both IMC semantics are equivalent with respect to quantitative reachability, which allows the extension of all results from [20, 5] to the at-every-step semantics. Based on this result, we also extend the CSP encodings introduced in Section 4 in order to solve quantitative reachability properties on pIMCs regardless of their semantics.

5.1 Equivalence of $\models^o_{\mathrm{I}}$ and $\models^a_{\mathrm{I}}$ w.r.t quantitative reachability

Given an IMC $\mathcal{I} = (S, s_0, P, V)$ and a state label $\alpha \subseteq A$, a quantitative reachability property on $\mathcal{I}$ is a property of the type $\mathbb{P}^{\mathcal{I}}(\Diamond\alpha) \sim p$, where $0 < p < 1$ and $\sim \in \{\leq, <, >, \geq\}$. Such a property is verified iff there exists an MC $\mathcal{M}$ satisfying $\mathcal{I}$ (with the chosen semantics) such that $\mathbb{P}^{\mathcal{M}}(\Diamond\alpha) \sim p$.

As explained above, all existing techniques and tools for verifying quantitative reachability properties on IMCs only focus on the once-and-for-all semantics. Indeed, in this setting, quantitative reachability properties are easier to compute because the underlying graph structure of all implementations is known. However, to the best of our knowledge, there are no works addressing the same problem with the at-every-step semantics or showing that addressing the problem in the once-and-for-all setting is sufficiently general. The following theorem fills this theoretical gap by proving that both semantics are equivalent w.r.t quantitative reachability. In other words, for all MC $\mathcal{M}$ such that $\mathcal{M} \models^a_{\mathrm{I}} \mathcal{I}$ and all state label α, there exist MCs $\mathcal{M}_{\leq}$ and $\mathcal{M}_{\geq}$ such that $\mathcal{M}_{\leq} \models^o_{\mathrm{I}} \mathcal{I}$, $\mathcal{M}_{\geq} \models^o_{\mathrm{I}} \mathcal{I}$ and $\mathbb{P}^{\mathcal{M}_{\leq}}(\Diamond\alpha) \leq \mathbb{P}^{\mathcal{M}}(\Diamond\alpha) \leq \mathbb{P}^{\mathcal{M}_{\geq}}(\Diamond\alpha)$. This is formalized in the following theorem.

Theorem 1. *Let $\mathcal{I} = (S, s_0, P, V)$ be an* IMC*, $\alpha \subseteq A$ be a state label, $\sim \in \{\leq, <, >, \geq\}$ and $0 < p < 1$. $\mathcal{I}$ satisfies $\mathbb{P}^{\mathcal{I}}(\Diamond\alpha) \sim p$ with the once-and-for-all semantics iff $\mathcal{I}$ satisfies $\mathbb{P}^{\mathcal{I}}(\Diamond\alpha) \sim p$ with the at-every-step semantics.*

The proof is constructive (see [4]): we use the structure of the relation $\mathcal{R}$ from the definition of $\models^a_{\mathrm{I}}$ in order to build the MCs $\mathcal{M}_{\leq}$ and $\mathcal{M}_{\geq}$.

5.2 Constraint Encodings

Note that the result from Theorem 1 naturally extends to pIMCs. We therefore exploit this result to construct a CSP encoding for verifying quantitative reachability properties in pIMCs. As in Section 4, we extend the CSP $\mathbf{C}_{\exists\mathbf{c}}$, that produces a correct MC implementation for the given pIMC, by imposing that this MC implementation satisfies the given quantitative reachability property. In order to compute the probability of reaching state label α at the MC level, we use standard techniques from [2] that require the partitioning of the state space into three sets $S_{\top}$, $S_{\bot}$, and $S_{?}$ that correspond to states reaching α with probability 1, states from which α cannot be reached, and the remaining states, respectively. Once this partition is chosen, the reachability probabilities of all states in $S_{?}$ are computed as the unique solution of a linear equation system (see [2], Theorem 10.19, p.766). We now explain how we identify states from $S_{\bot}, S_{\top}$ and $S_{?}$ and how we encode the linear equation system, which leads to the resolution of quantitative reachability.

Let $\mathcal{P} = (S, s_0, P, V, Y)$ be a pIMC and $\alpha \subseteq A$ be a state label. We start by setting $S_{\top} = \{s \mid V(s) = \alpha\}$. We then extend $\mathbf{C}_{\exists\mathbf{r}}(\mathcal{P})$ in order to identify the set $S_{\bot}$. Let $\mathbf{C}'_{\exists\mathbf{r}}(\mathcal{P}, \alpha) = (X \cup X', D \cup D', C \cup C')$ be such that $(X, D, C) = \mathbf{C}_{\exists\mathbf{r}}(\mathcal{P})$, X' contains one Boolean variable λ_s and one integer variable α_s with domain $[0, |S|]$ per state s in S, D' contains the domains of these variables, and C' is composed of the following constraints for each state $s \in S$:

(11) $\alpha_s = 1$, if $\alpha = V(s)$ **(12)** $\alpha_s \neq 1$, if $\alpha \neq V(s)$ **(13)** $\lambda_s \Leftrightarrow (\rho_s \wedge (\alpha_s \neq 0))$

(14) $\alpha_s > 1 \Rightarrow \bigvee_{s' \in \texttt{Succ}(s) \setminus \{s\}} (\alpha_s = \alpha_{s'} + 1) \wedge (\theta_s^{s'} > 0)$, if $\alpha \neq V(s)$

(15) $\alpha_s = 0 \Leftrightarrow \bigwedge_{s' \in \texttt{Succ}(s) \setminus \{s\}} (\alpha_{s'} = 0) \vee (\theta_s^{s'} = 0)$, if $\alpha \neq V(s)$

Note that variables α_s play a symmetric role to variables ω_s from $\mathbf{C}_{\exists \mathbf{r}}$: instead of indicating the existence of a path from s_0 to s, they characterize the existence of a path from s to a state labeled with α. In addition, due to Constraint (**13**), variables λ_s are set to true iff there exists a path with non zero probability from the initial state s_0 to a state labeled with α passing by s. Thus, α cannot be reached from states s.t. $\lambda_s = \mathsf{false}$. Therefore, $S_\perp = \{s \mid \lambda_s = \mathsf{false}\}$.

Finally, we encode the equation system from [2] in a last CSP encoding that extends $\mathbf{C}'_{\exists \mathbf{r}}$. Let $\mathbf{C}_{\exists \bar{\mathbf{r}}}(\mathcal{P}, \alpha) = (X \cup X', D \cup D', C \cup C')$ be such that $(X, D, C) = \mathbf{C}'_{\exists \mathbf{r}}(\mathcal{P}, \alpha)$, X' contains one variable π_s per state s in S with domain $[0, 1]$, D' contains the domains of these variables, and C' is composed of the following constraints for each state $s \in S$:

(16) $\neg\lambda_s \Rightarrow \pi_s = 0$ **(17)** $\lambda_s \Rightarrow \pi_s = 1$, if $\alpha = V(s)$

(18) $\lambda_s \Rightarrow \pi_s = \Sigma_{s' \in \texttt{Succ}(s)} \pi_{s'} \theta_{s'}^s$, if $\alpha \neq V(s)$

As a consequence, variables π_s encode the probability with which state s eventually reaches α when s is reachable from the initial state and 0 otherwise.

Let $p \in [0, 1] \subseteq \mathbb{R}$ be a probability bound. Adding the constraint $\pi_{s_0} \leq p$ (resp. $\pi_{s_0} \geq p$) to the previous $\mathbf{C}_{\exists \bar{\mathbf{r}}}$ encoding allows to determine if there exists a MC $\mathcal{M} \models^a_{\texttt{pI}} \mathcal{P}$ such that $\mathbb{P}^{\mathcal{M}}(\Diamond\alpha) \leq p$ (resp $\geq p$). Formally, let $\sim \in \{\leq, <, \geq, >\}$ be a comparison operator, we write $\not\sim$ for its negation (*e.g.*, $\not\leq$ is $>$). This leads to the following theorem.

Theorem 2. *Let $\mathcal{P} = (S, s_0, P, V, Y)$ be a* pIMC*, $\alpha \subseteq A$ be a label, $p \in [0, 1]$, $\sim \in \{\leq, <, \geq, >\}$ be a comparison operator, and (X, D, C) be $\mathbf{C}_{\exists \bar{\mathbf{r}}}(\mathcal{P}, \alpha)$:*

- CSP $(X, D, C \cup (\pi_{s_0} \sim p))$ *is satisfiable iff* $\exists \mathcal{M} \models^a_{\texttt{pI}} \mathcal{P}$ *s.t.* $\mathbb{P}^{\mathcal{M}}(\Diamond\alpha) \sim p$
- CSP $(X, D, C \cup (\pi_{s_0} \not\sim p))$ *is unsatisfiable iff* $\forall \mathcal{M} \models^a_{\texttt{pI}} \mathcal{P}$*:* $\mathbb{P}^{\mathcal{M}}(\Diamond\alpha) \sim p$

6 Prototype Implementation

Our results have been implemented in a prototype tool[6] which generates the above CSP encodings, and CSP encodings from [11] as well. Given a pIMC in a text format inspired from [20], our tool produces the desired CSP as a SMT instance with the QF_NRA logic (Quantifier Free Non linear Real-number Arithmetic). This instance can then be fed to any solver accepting the SMT-LIB format with QF_NRA logic [3]. For our benchmarks, we chose Z3 [8] (latest version: 4.5.0).

QF_NRA does not deal with integer variables. In practice, logics mixing integers and reals are harder than those over reals only. Thus we obtained better results by encoding integer variables into real ones. In our implementations each

[6] All resources, benchmarks, and source code are available online as a Python library at `https://github.com/anicet-bart/pimc_pylib`

Benchmark	pIMC			$\mathbf{C}_{\exists \mathbf{c}}$			$\mathbf{C}_{\exists \mathbf{r}}$			$\mathbf{C}_{\exists \overline{\mathbf{r}}}$		
	#states	#trans.	#par.	#var.	#cstr.	time	#var.	#cstr.	time	#var.	#cstr.	time
NAND K=1; N=2	104	147	4	255	1,526	0.17s	170	1,497	0.19s	296	2,457	69.57s
NAND K=1; N=3	252	364	5	621	3,727	0.24s	406	3,557	0.30s	703	5,828	31.69s
NAND K=1; N=5	930	1,371	7	2,308	13,859	0.57s	1,378	12,305	0.51s	2,404	20,165	T.O.
NAND K=1; N=10	7,392	11,207	12	18,611	111,366	9.46s	9,978	89,705	13.44s	17,454	147,015	T.O.

Table 1: Benchmarks

integer variable x is declared as a real variable whose real domain bounds are its original integer domain bounds; we also add the constraint $x < 1 \Rightarrow x = 0$. Since we only perform incrementation of x this preserves the same set of solutions.

In order to evaluate our prototype, we extend the NAND model from [17][7]. The original MC NAND model has already been extended as a pMC in [9], where the authors consider a single parameter p for the probability that each of the N *nand* gates fails during the multiplexing. We extend this model to pIMC by considering intervals for the probability that the initial inputs are stimulated and we have one parameter per *nand* gate to represent the probability that it fails. pIMCs in text format are automatically generated from the PRISM model.

Table 1 summarizes the size of the considered instances of the model (in terms of states, transitions, and parameters) and of the corresponding CSP problems (in terms of number of variables and constraints). In addition, we also present the resolution time of the given CSPs using the Z3 solver. Our experiments were performed on a 2.4 GHz Intel Core i5 processor with time out set to 10 minutes and memory out set to 2Gb.

7 Conclusion and future work

In this paper, we have compared several Markov Chain abstractions in terms of succinctness and we have shown that Parametric Interval Markov Chain is a strictly more succinct abstraction formalism than other existing formalisms such as Parametric Markov Chains and Interval Markov Chains. In addition, we have proposed constraint encodings for checking several properties over pIMC. In the context of qualitative properties such as existencial consistency or consistent reachability, the size of our encodings is significantly smaller than other existing solutions. In the quantitative setting, we have compared the two usual semantics for IMCs and pIMCs and showed that both semantics are equivalent with respect to quantitative reachability properties. As a side effect, this result ensures that all existing tools and algorithms solving reachability problems in IMCs under the once-and-for-all semantics can safely be extended to the at-every-step semantics with no changes. Based on this result, we have then proposed CSP encodings addressing quantitative reachability in the context of pIMCs regardless of the chosen semantics. Finally, we have developed a prototype tool that automatically generates our CSP encodings and that can be plugged to any constraint solver accepting the SMT-LIB format as input.

We plan to develop our tool for pIMC verification in order to manage other, more complex, properties (*e.g.*, supporting the LTL-language in the spirit of what

[7] Available online at `http://www.prismmodelchecker.com`

Tulip [20] does). We also plan on investigating a practical way of computing and representing the set of *all solutions* to the parameter synthesis problem.

References

1. Alur, R., Henzinger, T.A., Vardi, M.Y.: Parametric real-time reasoning. In: STOC. pp. 592–601. ACM (1993)
2. Baier, C., Katoen, J.P.: Principles of Model Checking (Representation and Mind Series). The MIT Press (2008)
3. Barrett, C., Fontaine, P., Tinelli, C.: The Satisfiability Modulo Theories Library (SMT-LIB). `www.SMT-LIB.org` (2016)
4. Bart, A., Delahaye, B., Lime, D., Monfroy, E., Truchet, C.: Reachability in Parametric Interval Markov Chains using Constraints (2017), `https://hal.archives-ouvertes.fr/hal-01529681`, long version
5. Benedikt, M., Lenhardt, R., Worrell, J.: LTL model checking of interval Markov chains. In: International Conference on Tools and Algorithms for the Construction and Analysis of Systems. pp. 32–46. Springer (2013)
6. Cantor, G.: ber unendliche, lineare punktmannigfaltigkeiten v (on infinite, linear point-manifolds). In: Mathematische Annalen, vol. 21, p. 545 591 (1883)
7. Courcoubetis, C., Yannakakis, M.: The complexity of probabilistic verification. J. ACM 42(4), 857–907 (1995)
8. De Moura, L., Bjørner, N.: Z3: An efficient SMT solver. In: TACAS/ETAPS. pp. 337–340. Springer (2008)
9. Dehnert, C., Junges, S., Jansen, N., Corzilius, F., Volk, M., Bruintjes, H., Katoen, J.P., Ábrahám, E.: PROPhESY: A PRObabilistic ParamEter SYnthesis Tool, pp. 214–231. Springer International Publishing, Cham (2015)
10. Delahaye, B.: Consistency for parametric interval Markov chains. In: SynCoP. pp. 17–32 (2015)
11. Delahaye, B., Lime, D., Petrucci, L.: Parameter synthesis for parametric interval Markov chains. In: VMCAI. pp. 372–390 (2016)
12. Gribaudo, M., Manini, D., Remke, A. (eds.): Model Checking of Open Interval Markov Chains. Springer, Cham (2015)
13. Hahn, E.M., Hermanns, H., Wachter, B., Zhang, L.: PARAM: A model checker for parametric Markov models. In: CAV. LNCS, vol. 6174. Springer (2010)
14. Husmeier, D., Dybowski, R., Roberts, S.: Probabilistic Modeling in Bioinformatics and Medical Informatics. Springer Publishing Company, Incorporated (2010)
15. Jonsson, B., Larsen, K.G.: Specification and refinement of probabilistic processes. In: LICS. pp. 266–277 (1991)
16. Kwiatkowska, M.Z., Norman, G., Parker, D.: PRISM 4.0: Verification of probabilistic real-time systems. In: CAV. LNCS, vol. 6806, pp. 585–591. Springer (2011)
17. Norman, G., Parker, D., Kwiatkowska, M., Shukla, S.: Evaluating the reliability of NAND multiplexing with PRISM. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 24(10), 1629–1637 (2005)
18. Puggelli, A., Li, W., Sangiovanni-Vincentelli, A.L., Seshia, S.A.: Polynomial-time verification of pctl properties of mdps with convex uncertainties. In: CAV. pp. 527–542. Springer (2013)
19. Rossi, F., Beek, P.v., Walsh, T.: Handbook of Constraint Programming (Foundations of Artificial Intelligence). Elsevier Science Inc. (2006)
20. Wongpiromsarn, T., Topcu, U., Ozay, N., Xu, H., Murray, R.M.: Tulip: A software toolbox for receding horizon temporal logic planning (2011)