Theoretical Computer Science 412 (2011) 4373–4404

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Constraint Markov Chains[☆]

Benoît Caillaud[a], Benoît Delahaye[b], Kim G. Larsen[c], Axel Legay[a,*], Mikkel L. Pedersen[c], Andrzej Wąsowski[d]

[a] *INRIA/IRISA, Rennes, France*
[b] *Université de Rennes 1/IRISA, Rennes, France*
[c] *Aalborg University, Denmark*
[d] *IT University of Copenhagen, Denmark*

A R T I C L E   I N F O

Article history:
Received 18 July 2010
Received in revised form 19 April 2011
Accepted 5 May 2011
Communicated by J. Esparza

Keywords:
Specification theory
Markov Chains
Compositional reasoning
Abstraction
Process algebra

A B S T R A C T

Notions of specification, implementation, satisfaction, and refinement, together with operators supporting stepwise design, constitute a specification theory. We construct such a theory for Markov Chains (MCs) employing a new abstraction of a Constraint MC. Constraint MCs permit rich constraints on probability distributions and thus generalize prior abstractions such as Interval MCs. Linear (polynomial) constraints suffice for closure under conjunction (respectively parallel composition). This is the first specification theory for MCs with such closure properties. We discuss its relation to simpler operators for known languages such as probabilistic process algebra. Despite the generality, all operators and relations are computable.

1. Introduction

Compositional design [1] is a research field, which aims at the development of mathematical foundations for reasoning about components. Usually this is achieved by specifying and analyzing the interfaces of components in order to infer global properties of a system in an incremental way. One popular approach in this area is the work on type systems, and in particular, on type systems for modules in the programming language community. Another approach, in the verification area, is the work on *specification theories*, which provide a modeling language for designing, evolving and advisedly reusing components with formal guarantees. For example, a large system, or a complex communication protocol, can be designed *stepwise* – by building more and more refined models – and analyzed *piecewise* by analyzing fragments of models and reasoning about the properties of the parallel composition. The hope is that this way we can combat the size and complexity of modern systems.

For functional analysis of discrete-time non-probabilistic systems, the theory of Modal Transition Systems (MTSs) [2,3] provides a specification formalism supporting refinement as well as conjunction and parallel composition. It has been recently applied to construct interface theories [4,5], which are extensions of classical interface automata proposed by de Alfaro et al. [6–10].

[☆] A preliminary version of this paper has appeared in the 7th International Conference on Quantitative Evaluation of Systems.
[*] Corresponding author. Tel.: +32 476 27 57 12.
E-mail addresses: benoit.caillaud@irisa.fr (B. Caillaud), benoit.delahaye@irisa.fr (B. Delahaye), kgl@cs.aau.dk (K.G. Larsen), alegay@irisa.fr, axel.legay@irisa.fr (A. Legay), mikkelp@cs.aau.dk (M.L. Pedersen), wasowski@itu.dk (A. Wąsowski).

0304-3975/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2011.05.010

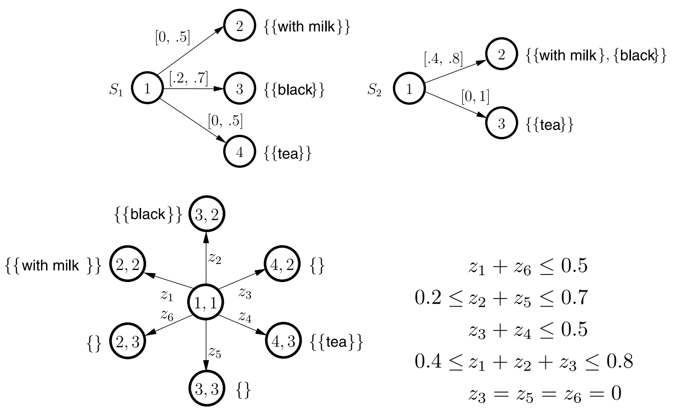


Fig. 1. IMCs showing non-closure under conjunction. (Top) The two specifications of different aspects of a coffee service. (Bottom) A conjunction expressed as a Markov Chain with linear constraints over probability values.

As soon as systems include randomized algorithms, probabilistic protocols, or interact with physical environment, probabilistic models are required to reason about them. This is exacerbated by requirements for fault tolerance, when systems need to be analyzed quantitatively for the amount of failure they can tolerate, or for the delays that may appear. As Henzinger and Sifakis [1] point out, introducing probabilities into design theories allows assessing dependability of IT systems in the same manner as commonly practiced in other engineering disciplines.

Generalizing the notion of MTSs to the non-functional analysis of probabilistic systems, the formalism of Interval Markov Chains (IMCs) was introduced in [11]; with notions of satisfaction and refinement generalizing probabilistic bisimulation. Informally, IMCs extend Markov Chains by labeling transitions with *intervals* of allowed probabilities rather than concrete probability values. Implementations of IMCs are Markov Chains (MCs) whose probability distributions match the constraints induced by the intervals. IMCs are known to be an efficient model on which refinement checking can be performed with efficient algorithms from linear algebra. Unfortunately, as we shall now see, the expressive power of IMCs is inadequate to support both conjunction and parallel composition.

Consider the IMCs of Fig. 1. S_1 specifies a behavior of a user of a coffee machine. It prescribes that a typical user orders coffee with milk with probability within $[0, 0.5]$ and orders black coffee with probability in $[0.2, 0.7]$. Customers also buy tea with probability in the interval $[0, 0.5]$. Now the vendor of the machine delivers another specification, S_2, which prescribes that the machine is functioning only if coffee (white or black) is ordered with probability between 0.4 and 0.8. Otherwise, the machine runs out of coffee powder too frequently, or the powder becomes too old. A conjunction of these two models would describe users who have use patterns compatible with this particular machine. In the bottom part of Fig. 1 we present the structure of such a conjunction. States $(2, 3)$, $(3, 3)$, and $(4, 2)$ are inconsistent and thus the corresponding probabilities must be zero: $z_3 = z_5 = z_6 = 0$. Now, attempting to express the conjunction $S_1 \wedge S_2$ as an IMC by a simple intersection of bounds gives $0.4 \leq z_1 \leq 0.5$, $0.4 \leq z_2 \leq 0.7$, and $z_4 \leq 0.5$. However, this naive construction is too coarse: whereas $(z_1, z_2, z_3, z_4, z_5, z_6) = (0.5, 0.5, 0, 0, 0, 0)$ satisfies the constraints the resulting overall probability of reaching a state resulting from State 2 of S_2, i.e. $z_1 + z_2 + z_3 = 1$, violates the upper bound of 0.8 specified in S_2.

Instead the conjunction should require, among others, that $z_1 + z_2 + z_3 \in [0.4, 0.8]$, which is not an interval constraint. This can be seen be pointing out an extremal point, which is not a solution, while all its coordinates take part in some solution. The bottom right part of Fig. 1 lists all the needed constraints over z_i necessary to express conjunction. A similar example can show that IMCs are not closed under parallel composition, either.

One way to approach this problem could be to work with two types of specifications: IMCs for refinement, and with a probabilistic logic such as PCTL [12] on which a logical conjunction is naturally defined. Such a solution is clearly not satisfactory. Indeed, according to [13], there is no procedure to synthesize a MC (an implementation) that satisfies two PCTL formulas in the quantitative case. It is also not possible to structurally compose two logical PCTL formulas.

The solution promoted in this paper is to enrich the model of IMCs. More precisely, we introduce *Constraint Markov Chains* (CMCs) as a foundation for component-based design of probabilistic systems. CMCs are a further extension of IMCs allowing rich constraints on the next-state probabilities from any state. Whereas linear constraints suffice for closure under conjunction, polynomial constraints are necessary for closure under parallel composition. We provide constructs for refinement, consistency checking, conjunction *and* parallel composition of CMC specifications — all indispensable ingredients of a compositional design methodology.

Specification theories. Let us give an overview of specification theories from the point of view of the main operators, and how they are supposed to be used. A more detailed methodological presentation, using the example not of probabilistic, but of timed systems, is available in [14].

Consistency and satisfaction. The fundamental notion of a specification theory is *satisfaction*—a relation that binds the specifications to their realizations (*implementations* or models). In our case, specifications are Constraint Markov Chains:

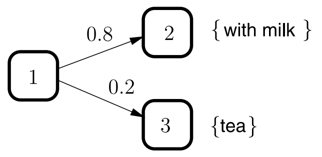


Fig. 2. A Markov Chain satisfying specification S_2 of Fig. 1.

mathematical over-approximations of probabilistic behaviors. Implementations are concrete random processes—Markov Chains.

We have shown two examples of IMC specifications in the top of Fig. 1: one of expectations of use (S_2) and one of usage in practice (S_1). Observe that these IMCs are also CMCs. In both cases it was easy to see that the specifications are *consistent*, i.e. for each of them one can derive a Markov Chain which satisfies all the interval constraints. For IMC S_2, the example of such an implementation is shown in Fig. 2. In development of probabilistic protocols, consistency means that given an abstract specification of a protocol, it is possible to derive a concrete random process satisfying its constraints.

It should be decidable whether a specification admits at least one implementation, and whether a system implements a specification. In our theory, an implementation shall not be viewed as a program in a general purpose programming language but rather as a mathematical object that represents a set of programs sharing common control properties.

Refinement. A *refinement* relation allows to explain whether a given specification S is a proper, consistent elaboration of another more abstract specification T. If this is the case then every implementation of S would also be admitted by T—none of such implementations would violate any requirements of T.

In system development the refinement relation is used to express correctness of a stepwise process, when more coarse-grained descriptions are refined into more detailed ones, until an implementation is obvious. Dually, in verification, refinement is used to establish that an abstract specification is refined by our concrete model. Then the abstract specification, which is usually smaller, can be more efficiently verified for properties preserved by refinement.

In Fig. 1 specification S_2 does not refine the specification S_1. This is witnessed by the Markov Chain in Fig. 2, which implements S_2, but not S_1.

Parallel composition. A theory should provide a combination operator on specifications, reflecting the standard composition of systems by putting different components together. In the coffee machine example such a situation could arise, if we wanted to analyze the model of a given coffee machine (not shown here), and a given customer, to see whether these two models are compatible. The first step of such an analysis would include combining the two models using parallel composition. Similarly, in protocol development, parallel composition can be used to combine, for example, specifications of a client and a server.

It is a common modeling scenario to use parallel composition of components, say S_1 and S_2, to build specifications of systems that are supposed to refine general requirements specified using one model, for example T. Then refinement check is performed, $S_1 \parallel S_2 \leq T$, to verify whether indeed a decomposition into components is correct.

Conjunction. In contrast to parallel composition, conjunction is used to combine different specifications for the very same component. Such a need could arise, if the model of the component consists of several separate specifications for different viewpoints, or arising from different stakeholders. The conjunction of two specifications should thus correspond to a specification whose implementations are all implementations of both conjoined specifications.

In our example, we already know that not all implementations of the coffee machine (S_2) are also implementations of S_1—in practical terms there exist machines that have requirements incompatible with requirements of our users. A different question is to ask, whether there exist any machines that are able to work with at least one of our users. This corresponds to asking whether specification $S_1 \wedge S_2$ is consistent. It is not hard to check that this specification, presented as a CMC in Fig. 1, can be satisfied by a Markov Chain, as already discussed.

Incremental design. A theory should allow incremental design (composing or conjoining specifications in any order) and independent implementability (composable specifications can always be refined separately) [15].

For example it should be possible to create specifications for communicating components S_1 and S_2, and refining them independently, to say P_1 and P_2 respectively, without loosing the overall refinement; so still $P_1 \parallel P_2 \leq S_1 \parallel S_2$.

Detailed results. In the above summary we illustrated the main operators of a specification theory using the IMCs of Figs. 1 and 2. However, as argued earlier, such a theory cannot be build for IMCs due to the lack of suitable closure properties. In this paper we develop the theory for Constraint Markov Chains. Below we summarize the most important design decisions. A less experienced reader may choose to skip this rather technical summary during the first reading.

The notions of satisfaction and strong/weak refinements for CMCs conservatively extend similar notions for IMCs [16,11]. We characterize these relations in terms of implementation set inclusion. In particular, in the main theorem, we prove that for deterministic CMCs weak and strong refinements are complete with respect to implementation set inclusion. In addition, we provide a construction, which for any CMC S returns a deterministic CMC $\varrho(S)$ containing the models of S. Refinement relations are not complete for non-deterministic CMCs, but one can show that the weak refinement is more likely to coincide with implementation set inclusion in such a context. We show that refinement between CMCs with polynomial constraints can be decided in essentially single exponential time.

In CMCs, each state is also labeled with a set of subsets of atomic propositions. Those propositions represent properties that should be satisfied by the implementation, the idea being that the satisfaction relation ensures that an implementation matches at least one of the subsets. This allows the specification to make additional abstraction of the behaviors of the implementation. Hence, at the level of specification, our model presents choices on subsets of atomic propositions. However these choices are independent from the probabilistic ones in the sense that any CMC whose states are labeled with a set of subsets of atomic propositions can be turned to an equivalent (in terms of set of implementations) CMC whose states are labeled with a single subset of atomic propositions. There, choices between the subsets of atomic propositions disappear. It is thus not surprising that our notion of parallel composition is following the widely accepted *principle of separation of concerns*. The idea is to separate parallel composition of probability distributions from synchronization on sets of atomic propositions. This separation can be found in probabilistic specification theories that have probabilistic automata as an underlying semantic model [17–20]. In fact, we show how probabilistic automata can be represented as CMCs, and how the traditional notions of parallel composition on such a model can be derived in our framework. This latter result shows that CMCs capture computational structure of known models and operators, laying down a basis for studying shared properties of many probabilistic automata based languages. We exemplify this by showing how precongruence properties for composition of probabilistic automata and known refinements can be obtained by reductions to CMCs.

We also compare the expressiveness of the operation of parallel composition and the one of conjunction. It turns out that for independent sets of valuations, composition refines conjunction, but the opposite is not true. This result allows to isolate a class of CMCs and CMCs operations that is closed under linear constraints. Finally, we also show that CMCs are generally not closed under disjunction and we discuss the problem of deciding whether a CMC is universal.

Structure of the paper. The paper is structured as follows. In Section 3, we introduce the concept of CMCs, satisfaction relation with respect to Markov Chains and the problem of consistency. Refinement is discussed in Section 4 while conjunction is presented in Section 5. Parallel composition is introduced in Section 6, where we also compare the operation to conjunction. Disjunction and universality are discussed in Section 7. In Section 8, we introduce deterministic CMCs and show that, for this class of CMCs, strong and weak refinements coincide with inclusion of implementation sets. Section 9 discusses the class of polynomial CMCs, which is the smallest class of CMCs closed under all the compositional design operations. Section 11 concludes the paper with related and future work.

2. Background definitions

In this section, we introduce concepts and definitions that will be used throughout the rest of the paper.

Let A, B be sets of propositions with $A \subseteq B$. The *restriction of* $W \subseteq B$ *to* A is given by $W\downarrow_A \equiv W \cap A$. If $T \subseteq 2^B$, then $T\downarrow_A \equiv \{W\downarrow_A \mid W \in T\}$. For $W \subseteq A$ define the *extension of* W *to* B as $W\uparrow^B \equiv \{V \subseteq B \mid V\downarrow_A = W\}$, so the set of sets whose restriction to A is W. Lift it to sets of sets as follows: if $T \subseteq 2^A$, then $T\uparrow^B \equiv \{W \subseteq B \mid W\downarrow_A \in T\}$.

Let $M, \Delta \in [0,1]^{n\times k}$ be two matrices and $x \in [0,1]^k$ be a row vector. We write M_{ij} for the cell in ith row and jth column of M, M_p for the pth row of M, and x_i for the ith element of x. Finally, Δ is a *correspondence matrix* iff $0 \le \sum_{j=1}^{k} \Delta_{ij} \le 1$ for all $1 \le i \le n$. We define the following operations:

1. If $\Delta \in [0,1]^{k\times q}$ and $\Delta' \in [0,1]^{k\times r}$ are two correspondence matrices, we define $\Delta'' = \Delta \otimes \Delta'$ by $\Delta'' \in [0,1]^{k\times(qr)}$ and $\Delta''_{i(j,n)} = \Delta_{ij}\Delta'_{in}$.
2. If $\Delta \in [0,1]^{k\times q}$ and $\Delta' \in [0,1]^{r\times s}$ are two correspondence matrices, we define $\Delta'' = \Delta \odot \Delta'$ by $\Delta'' \in [0,1]^{(kr)\times(qs)}$ and $\Delta''_{(i,j)(n,p)} = \Delta_{in}\Delta'_{jp}$.

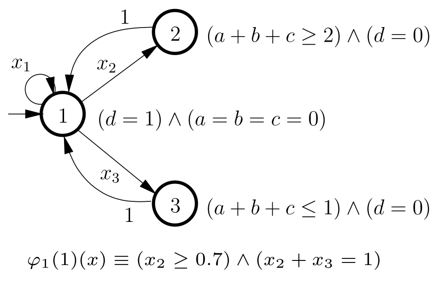
We have the following lemma.

Lemma 1. 1. *Let $\Delta \in [0,1]^{k\times q}$ and $\Delta' \in [0,1]^{q\times r}$ be two correspondence matrices. The matrix $\Delta'' = \Delta\Delta'$ is a correspondence matrix;*
2. *Let $\Delta \in [0,1]^{k\times q}$ and $\Delta' \in [0,1]^{k\times r}$ be two correspondence matrices. The matrix $\Delta'' = \Delta \otimes \Delta'$ is a correspondence matrix;*
3. *Let $\Delta \in [0,1]^{k\times q}$ and $\Delta' \in [0,1]^{r\times s}$ be two correspondence matrices. The matrix $\Delta'' = \Delta \odot \Delta'$ is a correspondence matrix;*

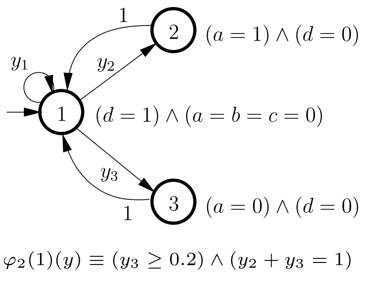
Proof. We first prove the upper bound on the row-sum of each of the matrices.

1. Let $1 \le i \le k$ and $1 \le j \le r$. We have $\Delta''_{ij} = \sum_{n=1}^{q} \Delta_{in}\Delta'_{nj}$. Thus,

$$\begin{aligned}\sum_{j=1}^{r} \Delta''_{ij} &= \sum_{j=1}^{r}\sum_{n=1}^{q} \Delta_{in}\Delta'_{nj} = \sum_{n=1}^{q}\sum_{j=1}^{r} \Delta_{in}\Delta'_{nj} \\ &= \sum_{n=1}^{q}\left(\Delta_{in}\sum_{j=1}^{r}\Delta'_{nj}\right) \le \sum_{n=1}^{q}\Delta_{in} \le 1.\end{aligned}$$


1
x_1
x_2
x_3
1
2
3
1
$(a+b+c \geq 2) \wedge (d=0)$
$(d=1) \wedge (a=b=c=0)$
$(a+b+c \leq 1) \wedge (d=0)$
$\varphi_1(1)(x) \equiv (x_2 \geq 0.7) \wedge (x_2+x_3=1)$

(a) CMC S_1, the customer specification of the optical relay.



(b) CMC S_2, the manufacturer specification of the relay.

Fig. 3. Two CMCs specifying an optical relay from different perspectives.

2. Let $1 \leq i \leq k$ and $(j, n) \in \{1, \ldots, q\} \times \{1, \ldots, r\}$. We have $\Delta''_{i(j,n)} = \Delta_{ij}\Delta'_{in}$. Thus, similarly as above:
$$\sum_{(j,n)\in\{1,\ldots q\}\times\{1,\ldots r\}} \Delta''_{i(j,n)} = \sum_{j=1}^{q}\sum_{n=1}^{r} \Delta_{ij}\Delta'_{in} = \sum_{j=1}^{q}\left(\Delta_{ij}\sum_{n=1}^{r}\Delta'_{in}\right) \leq 1.$$
3. Let $(i, j) \in \{1, \ldots k\} \times \{1, \ldots r\}$ and $(n, p) \in \{1, \ldots q\} \times \{1, \ldots s\}$. We have $\Delta''_{(i,j)(n,p)} = \Delta_{in}\Delta'_{jp}$. Thus,
$$\sum_{(n,p)\in\{1,\ldots q\}\times\{1,\ldots s\}} \Delta''_{(i,j)(n,p)} = \sum_{n=1}^{q}\sum_{p=1}^{s} \Delta_{in}\Delta'_{jp} = \left(\sum_{n=1}^{q}\Delta_{in}\right)\left(\sum_{p=1}^{s}\Delta'_{jp}\right) \leq 1.$$

All three row-sums are non-negative, as all the matrices only contain non-negative numbers. □

3. Constraint Markov chains

In this section, we explicitly introduce the concept of *Constraint Markov Chains* (CMCs). We first begin with the definition of *Markov Chains* (MCs) that act as models for CMCs.

Definition 2 (*Markov Chain*)**.** $P = \langle\{1, \ldots, n\}, o, M, A, V\rangle$ is a *Markov Chain* if $\{1, \ldots, n\}$ is a set of states containing the initial state o, A is a set of atomic propositions, $V : \{1, \ldots, n\} \to 2^A$ is a state valuation, and $M \in [0, 1]^{n\times n}$ is a probability transition matrix: $\sum_{j=1}^{n} M_{ij} = 1$ for $i = 1, \ldots, n$.

We now introduce *Constraint Markov Chains* (CMCs for short), a finite representation for a possibly infinite set of MCs. Roughly speaking, CMCs generalize MCs in that, instead of specifying a concrete transition matrix, they only constrain probability values in the matrix. Constraints are modeled using a *characteristic function*, which for a given source state and a distribution of probabilities of leaving the state evaluates to 1 iff the distribution is permitted by the specification. Similarly, instead of a concrete valuation function for each state, a *constraint on valuations* is used. Here, a valuation is permitted iff it is contained in the set of admissible valuations of the specification.

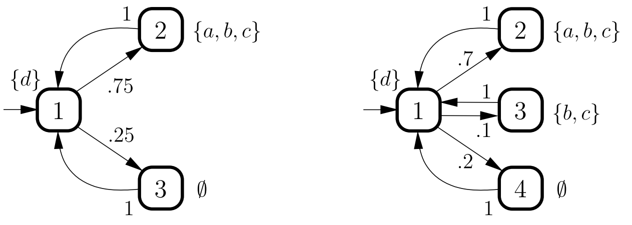
Definition 3 (*Constraint Markov Chain*)**.** A *Constraint Markov Chain* is a tuple $S = \langle\{1, \ldots, k\}, o, \varphi, A, V\rangle$, where $\{1, \ldots, k\}$ is a set of states containing the initial state o, A is a set of atomic propositions, $V : \{1, \ldots, k\} \to 2^{2^A}$ is a set of admissible state valuations and $\varphi : \{1, \ldots, k\} \to [0, 1]^k \to \{0, 1\}$ is a *constraint function* such that, for all states $1 \leq j \leq k$, if $\varphi(j)(x) = 1$, then the x vector is a probability distribution: $x \in [0, 1]^k$ and $\sum_{i=1}^{k} x_i = 1$.

In the rest of the document, we consider that the last constraint, $\sum_{i=1}^{k} x_i = 1$, is implicit and usually dropped in the examples.
An *Interval Markov Chain* (IMC for short) [11] is a CMC whose constraint functions are represented by intervals, so for all $1 \leq i \leq k$ there exist constants α_i, β_i such that, for all states $1 \leq j \leq k$, $\varphi(j)(x) = 1$ iff $\forall 1 \leq i \leq k,\ x_i \in [\alpha_i, \beta_i]$.

Example. Two parties, a customer and a vendor, are discussing a design of a relay for an optical telecommunication network. The relay is designed to amplify an optical signal transmitted over a long distance optical fiber. The relay should have several modes of operation, modeled by four dynamically changing properties and specified by atomic propositions a, b, c, and d:

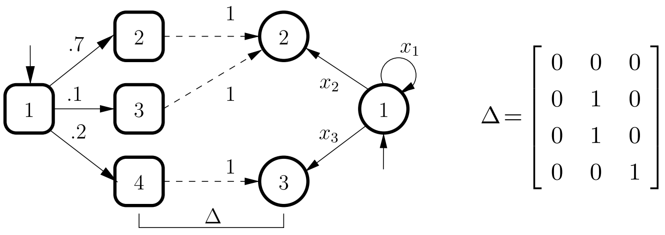
Atomic propositions in the optic relay specifications		
a	ber $\leq 10^{-9}$	Bit error rate lower than 1 per billion bits transmitted
b	br $>$ 10 Gbits/s	The bit-rate is higher than 10 Gbits/s.
c	$P <$ 10 W	Power consumption is less than 10 W.
d	Standby	The relay is not transmitting.

The customer presents CMC S_1 (Fig. 3a) specifying the admissible behavior of the relay from their point of view. States are labeled with formulas characterizing sets of valuations. For instance, "$(a + b + c \geq 2) \wedge (d = 0)$" at state 2 of S_1 represents



(a) Markov Chain P_1 satisfying S_1 and S_2. (b) Markov Chain P_2 satisfying S_1 and S_2.

Fig. 4. Two implementations (MCs) of an optical relay.



$$\Delta = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Fig. 5. Correspondence for initial states of P_2 and S_1.

$V_1(2) = \{\{a, b\}, \{b, c\}, \{a, c\}, \{a, b, c\}\}$, where a, b, c, and d range over Booleans. State 1 specifies a standby mode, where no signal is emitted and only marginal power is consumed. State 2 is the high power mode, offering a high signal/noise ratio, and hence a high bit-rate and low error rate, at the expense of a high power consumption. State 3 is the low power mode, with a low power consumption, low bit-rate and high error rate. The customer prescribes that the probability of the high power mode (state 2) is not less than 0.7. The vendor replies with CMC S_2 (Fig. 3b), which represents possible relays that they can build. Because of thermal limitations, the low power mode has a probability higher than 0.2.

A state u of S is (directly) *reachable* from a state i if there exists a probability distribution $x \in [0, 1]^k$ with a non-zero probability x_u, which satisfies $\varphi(i)(x)$.

3.1. Satisfaction

We relate CMC specifications to MCs implementing them by extending the definition of satisfaction presented in [11]. The main modification with regard to that definition is the matching of valuation constraints (instead of concrete valuations) and satisfying the full-fledged constraint functions of CMCs (instead of concrete probability distributions). Crucially, just like in [11], we abstract from syntactic structure of transitions—a single transition in the implementation MC can contribute to satisfaction of more than one transition in the specification by distributing its probability mass against several transitions. Similarly, several MC transitions can contribute to satisfaction of a single specification transition.

Definition 4 (*Satisfaction Relation*). Let $P = \langle\{1, \dots, n\}, o_P, M, A_P, V_P\rangle$ be a MC and $S = \langle\{1, \dots, k\}, o_S, \varphi, A_S, V_S\rangle$ be a CMC with $A_S \subseteq A_P$. Then $\mathcal{R} \subseteq \{1, \dots, n\} \times \{1, \dots, k\}$ is a *satisfaction relation* between states of P and S iff whenever $p\mathcal{R}u$, then

1. $V_P(p)\downarrow_{A_S} \in V_S(u)$, and
2. there exists a correspondence matrix $\Delta \in [0, 1]^{n\times k}$ such that
   - for all $1 \leq p' \leq n$ with $M_{pp'} \neq 0$, $\sum_{j=1}^{k} \Delta_{p'j} = 1$;
   - $\varphi(u)(M_p\Delta)$ holds and
   - if $\Delta_{p'u'} \neq 0$, then $p'\mathcal{R}u'$.

We write $P \models S$ iff there exists a satisfaction relation relating o_P and o_S, and call P an *implementation* of S. The set of all implementations of S is given by $[\![S]\!] \equiv \{P \mid P \models S\}$. Rows of Δ that correspond to reachable states of P always sum up to 1. This is to guarantee that the entire probability mass of implementation transitions is allocated. For unreachable states, we leave the corresponding rows in Δ unconstrained. P may have a richer set of atomic propositions than S, in order to facilitate abstract modeling: this way an implementation can maintain local information using internal variables. Algorithms to decide satisfaction are particular cases of algorithms to decide *refinement* between CMCs. See the next section.

Example. We illustrate the concept of correspondence matrix between Specification S_1 (given in Fig. 3a) and Implementation P_2 (given in Fig. 4b). The CMC S_1 has three outgoing transitions from state 1 but, due to constraint function in 1, the transition labeled with x_1 cannot be taken (the constraint implies $x_1 = 0$). The probability mass going from state 1 to states 2 and 3 in P_2 corresponds to the probability allowed by S_1 from its state 1 to its state 2; The redistribution is done with the help of the matrix Δ given in Fig. 5. The ith column in Δ describes how big a fraction of each transition probability

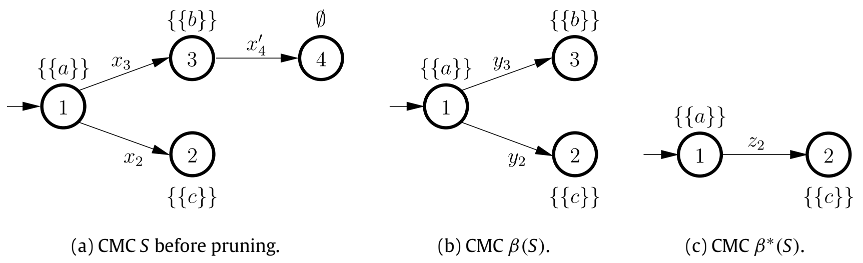


Fig. 6. Illustration of the pruning algorithm.

(for transitions leaving 1) is associated with probability x_i in S_1. Observe that the constraint function $\varphi_1(1)(0, 0.8, 0.2) = \varphi_1(1)((0, 0.7, 0.1, 0.2)\Delta)$ is satisfied.

CMC semantics follows the Markov Decision Process (MDP) tradition [21,22]. The MDP semantics is typically opposed to the Uncertain Markov Chain semantics, where the probability distribution from each state is fixed a priori.

States of CMCs are labeled with set of subsets of atomic propositions. A single set of propositions represents properties that should be satisfied by the implementation. A set of sets models a choice of properties, with the idea being that the satisfaction relation ensures that an implementation matches at least one of the subsets.

3.2. Consistency

We now define the notion of consistency and propose an algorithm that turns any consistent CMC in a CMC with no inconsistent states.

A CMC S is *consistent* if it admits at least one implementation. We now discuss how to decide consistency. A state u of S is *valuation consistent* iff $V(u) \neq \emptyset$; it is *constraint consistent* iff there exists a probability distribution vector $x \in [0, 1]^k$ such that $\varphi(u)(x) = 1$. It is easy to see that if *each state* of S is both valuation and constraint consistent, then S is also consistent. However, inconsistency of a state, called *local inconsistency*, does not imply inconsistency of the specification, called *global inconsistency*. Indeed, an inconsistent state could be made unreachable by forcing the probabilities to reach it to zero. The operations presented later in this paper may introduce inconsistent states, leaving a question if a resulting CMC is consistent. In order to decide whether S is inconsistent, state inconsistencies are propagated throughout the entire state-space using a *pruning operator* β that removes inconsistent states from S. The result $\beta(S)$ is a new CMC, which may still contain some inconsistent states. We define β formally. Let $S = \langle\{1, \ldots, k\}, o, \varphi, A, V\rangle$ be a CMC.

- If the initial state o is locally inconsistent, then let $\beta(S) = \emptyset$ (meaning that it is not well defined, returning an empty CMC).
- If S does not contain locally inconsistent states, then $\beta(S) = S$.
- Else proceed in two steps. Let $k' < k$ be the number of locally consistent states. Then define a function $\nu : \{1, \ldots, k\} \to \{\perp, 1, \ldots, k'\}$. All inconsistent states are mapped to $\perp$, i.e. for all $1 \le i \le k$ take $\nu(i) = \perp$ iff $[(V(i) = \emptyset) \vee (\forall x \in [0, 1]^k,\ \varphi(i)(x) = 0)]$. All remaining states are mapped injectively into $\{1, \ldots, k'\}$: $\nu(i) \neq \perp \implies \forall j \neq i,\ \nu(j) \neq \nu(i)$. Then let $\beta(S) = \langle\{1, \ldots, k'\}, \nu(o), \varphi', A, V'\}$, where $V'(i) = V(\nu^{-1}(i))$ and for all $1 \le j \le k'$ the constraint $\varphi'(j)(y_1, \ldots, y_{k'})$ is: $\exists x_1, \ldots, x_k$ such that
$$\left[\nu(q) = \perp \Rightarrow x_q = 0\right] \wedge \left[\forall 1 \le l \le k' : y_l = x_{\nu^{-1}(l)}\right] \wedge \left[\varphi(\nu^{-1}(j))(x_1, \ldots, x_k)\right].$$
The constraint makes the inconsistent states unreachable, and then $\perp$ is dropped as a state. Note that, in practice, the new constraint can be computed in linear time by applying substitutions (of zeros for inconsistent states probabilities) and variable renaming.

The operator is applied iteratively, until a fixpoint is reached. S is consistent if the resulting CMC $\beta^*(S)$ is non-empty. The unique maximum fixpoint is known to exist due to Tarski's theorem, as β is a monotonic decreasing operator on a finite powerset lattice ordered by set inclusion (β operates on sets of states). The following example illustrates the pruning algorithm.

Example. Consider the CMC $S = \langle\{1, 2, 3, 4\}, 1, \varphi, \{a, b, c\}, V\rangle$ given in Fig. 6a. Define φ as follows : $\varphi(1)(x) \equiv (x_3 \le 0.3) \wedge (x_2 + x_3 = 1)$, $\varphi(3)(x') \equiv (x'_4 = 1)$. The constraint of states 2 and 4 are not relevant for this example.

State 4 is obviously not valuation consistent. States 1, 2 and 3 are all valuations and constraint consistent. As a consequence, the first step of the pruning algorithm will only mark state 4 as inconsistent. Define the following function:
$$\nu = [1 \mapsto 1, 2 \mapsto 2, 3 \mapsto 3, 4 \mapsto \perp]. \tag{1}$$
Then define $\beta(S) = \langle\{1, 2, 3\}, 1, \varphi', \{a, b, c\}, V'\rangle$ such that, after reduction we have $\varphi'(1)(y) \equiv (y_3 \le 0.3) \wedge (y_2 + y_3 = 1)$, and $\varphi'(3)(y') \equiv \exists x'_4,\ (x'_4 = 0) \wedge (x'_4 = 1)$. $\beta(S)$ is given in Fig. 6b.

Obviously, state 3 of $\beta(S)$ is now constraint inconsistent: $\varphi'(3)(y')$ is not satisfiable. We thus apply another time the pruning operator β in order to remove state 3. This time we obtain a consistent CMC $\beta^*(S)$, given in Fig. 6c.

Theorem 5. *Let $S = \langle\{1,\ldots,k\}, o, \varphi, A, V\rangle\}$ be a CMC and let $\beta^*(S) = \lim_{n\to\infty}\beta^n(S)$ be the fixpoint of β. For any MC P, we have* (1) $P \models S \iff P \models \beta(S)$ *and* (2) $[\![S]\!] = [\![\beta^*(S)]\!]$.

Proof. Let $S = \langle\{1,\ldots,k\}, o, \varphi, A, V\rangle$ be a CMC (with at least one inconsistent state) and $P = \langle\{1,\ldots,n\}, o_P, M, A_P, V_P\rangle$ be a MC. Let $S' = \beta(S) = \langle\{1,\ldots,k'\}, o', \varphi', A, V'\rangle$ be the result of applying the pruning algorithm to S. If $\beta(S)$ is empty, then both S and $\beta(S)$ are inconsistent.

Consider a function ν for removing inconsistent states (one exists because there are inconsistent states), such that $k' < k$ and for all $1 \le i \le k$, $\nu(i) = \perp \iff [(V(i) = \emptyset) \vee (\forall x \in [0,1]^k,\ \neg\varphi(i)(x))]$ and $\nu(i) \neq \perp \Rightarrow \forall j \neq i,\ \nu(j) \neq \nu(i)$. We first show that $P \models S \iff P \models \beta(S)$.

$\Rightarrow$ Suppose that $P \models S$. Then there exists a satisfaction relation $\mathcal{R}$ such that $o_P \mathcal{R} o$. Define the relation $\mathcal{R}' \subseteq \{1,\ldots,n\} \times \{1,\ldots,k'\}$ such that $p\mathcal{R}'v$ iff there exists $u \in \{1,\ldots,k\}$ such that $p\mathcal{R}u$ and $\nu(u) = v$. It is clear that $o_P\mathcal{R}'o'$. We prove that $\mathcal{R}'$ is a satisfaction relation. Let p, u, v such that $p\mathcal{R}u$ and $\nu(u) = v$.

- As $\nu(u) \neq \perp$, we have by definition that $V'(v) = V(u)$, thus $V_P(p)\downarrow_A \in V'(v)$.
- Let $\Delta \in [0,1]^{n\times k}$ be the correspondence matrix witnessing $p\mathcal{R}u$. Let $\Delta' \in [0,1]^{n\times k'}$ such that $\Delta'_{qw} = \Delta_{q\nu^{-1}(w)}$. It is clear that Δ' is a correspondence matrix. We first show that
$$\forall u' \in \{1,\ldots,k\},\ (\nu(u') = \perp) \Rightarrow (\forall q \in \{1,\ldots,n\},\ \Delta_{qu'} = 0). \tag{2}$$
Let $u' \in \{1,\ldots,k\}$ such that $\nu(u') = \perp$, and suppose that there exists $q \in \{1,\ldots,n\}$, $\Delta_{qu'} \neq 0$. As Δ is a correspondence matrix, we have $q\mathcal{R}u'$. Thus $V_P(q)\downarrow_A \in V(u')$, which means that $V(u') \neq \emptyset$, and there exists Δ'' such that $\varphi(u')(M_q\Delta'')$. Thus, there exists $x \in [0,1]^k$ such that $\varphi(u')(x)$. As a consequence, we cannot have $\nu(u') = \perp$, which is a contradiction, thus (2).

  We now prove that $\mathcal{R}'$ satisfies the axioms of a satisfaction relation.

  1. Let $p' \in \{1,\ldots,n\}$ such that $M_{pp'} \neq 0$. This implies, by definition, that $\sum_{j=1}^{k}\Delta_{p'j} = 1$. We have $\sum_{j=1}^{k'}\Delta'_{p'j} = \sum_{r\in\{1,\ldots,k\}\mid\nu(r)\neq\perp}\Delta_{p'r}$. By (2), $\sum_{r\in\{1,\ldots,k\}\mid\nu(r)\neq\perp}\Delta_{p'r} = \sum_{r=1}^{k}\Delta_{p'r} = 1$.
  2. Let $y = M_p\Delta' \in [0,1]^{k'}$ and $x = M_p\Delta \in [0,1]^{1\times k}$. We know that $\varphi(u)(x)$ holds. Moreover, by (2), if $\nu(q) = \perp$, then $x_q = 0$, and for all $l \in \{1,\ldots,k'\}$, $y_l = x_{\nu^{-1}(l)}$. Clearly, this implies that $\varphi'(v)(M_p\Delta')$ holds.
  3. Let $p', v' \in \{1,\ldots,n\} \times \{1,\ldots,k'\}$ such that $\Delta'_{p'v'} \neq 0$. We have $\Delta'_{p'v'} = \Delta_{p'\nu^{-1}(v')} \neq 0$, thus there exists $u' \in \{1,\ldots,k\}$ such that $p'\mathcal{R}u'$ and $\nu(u') = v'$. Finally $p'\mathcal{R}'v'$.

  Finally, $\mathcal{R}'$ is a satisfaction relation such that $o_P\mathcal{R}'o'$, thus $P \models \beta(S)$.

$\Leftarrow$ Conversely, the reasoning is the same, except that we now build Δ from Δ' saying that $\Delta_{qv} = 0$ if $\nu(v) = \perp$ and $\Delta_{qv} = \Delta'_{q\nu(v)}$ otherwise.

We have proved that β is implementations-conservative. Thus the fixpoint of β verifies the same property (this can be concluded by mathematical induction, given that the fixpoint is always reached in a finite number of steps). □

The fixpoint of β, and thus the entire consistency check, can be computed using a quadratic number of state consistency checks. The complexity of each check depends on the constraint language that has been chosen.

3.3. Single valuation normal form

It turns out that any CMC whose states are labeled with a set of subsets of atomic propositions can be turned into an equivalent CMC (in terms of sets of implementations) whose states are labeled with sets that contains a single subset of atomic propositions. Hence, working with sets of subsets of valuations is a kind of modeling sugar that can be removed with a transformation to the *single valuation normal form*. We now give details regarding this theory.

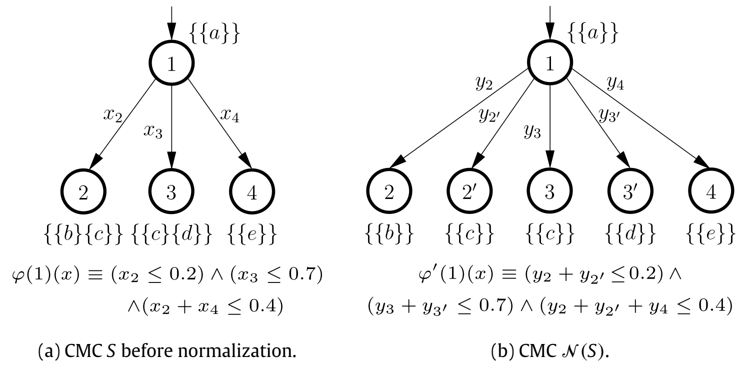
Definition 6. We say that a CMC is in a *Single Valuation Normal Form* if all its admissible valuation sets are singletons ($|V(i)| = 1$ for each $1 \le i \le k$).

More precisely every consistent CMC with at most one admissible valuation in the initial state can be transformed into the normal form preserving its implementation set.

The normalization algorithm, which is presented in Definition 7, basically separates each state u with m possible valuations into m states $u_1,\ldots,u_m$, each with a single admissible valuation. Then the constraint function is adjusted, by substituting sums of probabilities going to the new states in place of the old probabilities targeting u. The transformation is local and syntax based. It can be performed in polynomial time and it only increases the size of the CMC polynomially. We will write $\mathcal{N}(S)$ for a result of normalization of S.

Definition 7 (*Normalization Algorithm*). Let $S = \langle\{1,\ldots,k\}, o, \varphi, A, V\rangle$ be a CMC. The normalization of S is only defined if o is in single valuation normal form (i.e. $|V(o)| = 1$) and if there exists a function $\mathcal{N} : \{1,\ldots,k\} \to 2^{\{1,\ldots,m\}}$ such that:

1. $\{1,\ldots,m\} = \cup_{i\in\{1,\ldots,k\}}\mathcal{N}(i)$;



{{a}}
1
x_2
x_3
x_4
2
3
4
{{b}{c}} {{c}{d}} {{e}}
$\varphi(1)(x) \equiv (x_2 \le 0.2) \wedge (x_3 \le 0.7) \wedge (x_2 + x_4 \le 0.4)$
(a) CMC S before normalization.
{{a}}
1
y_2
$y_{2'}$
y_3
$y_{3'}$
y_4
2
2′
3
3′
4
{{b}} {{c}} {{c}} {{d}} {{e}}
$\varphi'(1)(x) \equiv (y_2 + y_{2'} \le 0.2) \wedge (y_3 + y_{3'} \le 0.7) \wedge (y_2 + y_{2'} + y_4 \le 0.4)$
(b) CMC $\mathcal{N}(S)$.

Fig. 7. Illustration of the normalization algorithm.

2. For all $1 \le i \ne j \le k$, $\mathcal{N}(i) \cap \mathcal{N}(j) = \emptyset$;
3. $\forall 1 \le i \le k, |\mathcal{N}(i)| = |V(i)|$;

Under these assumptions, the normalization of S is the CMC $\mathcal{N}(S) = \langle \{1, \ldots, m\}, o', \varphi', A, V' \rangle$ such that $\mathcal{N}(o) = o'$ and

1. $\forall 1 \le j \le m, |V'(j)| = 1$;
2. $\forall 1 \le i \le k, V(i) = \cup_{u \in \mathcal{N}(i)} V'(u)$;
3. $\forall 1 \le i \le k, \forall u, v \in \mathcal{N}(i), u \ne v \iff V'(u) \ne V'(v)$;
4. $\forall 1 \le j \le m. \varphi'(j)(x_1, \ldots x_m) = \varphi(\mathcal{N}^{-1}(j))(\sum_{u \in \mathcal{N}(1)} x_u, \ldots, \sum_{u \in \mathcal{N}(k)} x_u)$.

By construction, $\mathcal{N}(S)$ is in single valuation normal form. Moreover, if S is consistent, then a function $\mathcal{N}$ satisfying the conditions specified in Definition 7 exists.

The following example illustrates the normalization algorithm.

Example. Consider the CMC $S = \langle \{1, 2, 3, 4\}, 1, \varphi, \{a, b, c, d, e\}, V \rangle$ given in Fig. 7a. Since states 2 and 3 have two valuation sets, S is not in single valuation normal form. Define the following normalization function:

$$\mathcal{N} = \left[1 \mapsto \{1\}, 2 \mapsto \{2, 2'\}, 3 \mapsto \{3, 3'\}, 4 \mapsto 4\right]. \tag{3}$$

The result of applying the normalization algorithm to S is the CMC $\mathcal{N}(S) = \langle \{1, 2, 2', 3, 3', 4\}, 1, \varphi', \{a, b, c, d, e\}, V' \rangle$ given in Fig. 7b. Following the algorithms, states 2 and 3 of S have been each separated into two states with a single valuation. The constraint function of state 1 uses $y_2 + y_{2'}$ and $y_3 + y_{3'}$ instead of x_2 and x_3 respectively.

We now show that normalization preserves implementations of a CMC.

Theorem 8. *Let $S = \langle \{1, \ldots k\}, o, \varphi, A, V \rangle$ be a consistent CMC. If $|V(o)| = 1$, then for all MC P, we have $P \models S \iff P \models \mathcal{N}(S)$.*

Proof. Let $S = \langle \{1, \ldots, k\}, o, \varphi, A, V \rangle$ be a consistent CMC such that $|V(o)| = 1$. Let $S' = \mathcal{N}(S) = \langle \{1, \ldots, m\}, o', \varphi', A, V' \rangle$ and $\mathcal{N} : \{1, \ldots, k\} \to 2^{\{1, \ldots, m\}}$ be the associated function.
($\Rightarrow$) Let $P = \langle \{1, \ldots, n\}, o_P, M, A_P, V_P \rangle$ be a MC such that $P \models S$. Let $\mathcal{R}$ be the associated satisfaction relation. Let $\mathcal{R}' \subseteq \{1, \ldots, n\} \times \{1, \ldots, m\}$ be a new relation such that $p\mathcal{R}'u \iff V_P(p) \in V'(u)$ and $p\mathcal{R}\mathcal{N}^{-1}(u)$. We show that $\mathcal{R}'$ is a satisfaction relation. Let p, u such that $p\mathcal{R}'u$.

1. By definition, we have $V_P(p) \in V'(u)$.
2. We have $p\mathcal{R}\mathcal{N}^{-1}(u)$. Let $\Delta \in [0, 1]^{n \times k}$ be the associated correspondence matrix. Define $\Delta' \in [0, 1]^{n \times m}$ such that $\Delta'_{q,v} = \Delta_{q,\mathcal{N}^{-1}(v)}$ if $V_p(q) \in V'(v)$ and 0 else. As every coefficient of Δ appears once and only once in the same row of Δ', it is clear that Δ' is a correspondence matrix. Moreover,
   - If q is such that $M_{pq} \ne 0$, then $\sum_{j=1}^{m} \Delta'_{q,j} = \sum_{i=1}^{k} \Delta_{q,i} = 1$;
   - For all $1 \le i \le k$, $\sum_{j \in \mathcal{N}(i)} ([M_p \Delta']_j) = [M_p \Delta]_i$. As a consequence, $\varphi'(u)(M_p \Delta') = \varphi(\mathcal{N}^{-1}(u))(M_p \Delta)$ holds.
   - If q, v are such that $\Delta'_{q,v} \ne 0$, then $\Delta_{q,\mathcal{N}^{-1}(v)} \ne 0$ and $V_P(q) \in V'(v)$, thus $q\mathcal{R}'v$.

Finally, $\mathcal{R}'$ is a satisfaction relation. It is easy to see that $o_p\mathcal{R}'o'$. As a consequence, we have $P \models \mathcal{N}(S)$.
($\Leftarrow$) Let $P = \langle \{1, \ldots, n\}, o_P, M, A_P, V_P \rangle$ be a MC such that $P \models \mathcal{N}(S)$. Let $\mathcal{R}$ be the associated satisfaction relation. Let $\mathcal{R}' \subseteq \{1, \ldots, n\} \times \{1, \ldots, k\}$ such that $p\mathcal{R}'u \iff \exists j \in \mathcal{N}(u)$ s.t. $p\mathcal{R}j$. We will show that $\mathcal{R}'$ is a satisfaction relation. Let p, u such that $p\mathcal{R}'u$.

1. We have $V_P(p) \in V(u) = \cup_{j \in \mathcal{N}(u)} V'(j)$.
2. Let $j \in \mathcal{N}(u)$ such that $p\mathcal{R}j$, and let $\Delta \in [0, 1]^{n \times m}$ be the associated correspondence matrix. Define $\Delta' \in [0, 1]^{n \times k}$ such that $\Delta'_{q,v} = \sum_{i \in \mathcal{N}(v)} \Delta_{q,i}$. It is clear that for all q, $\sum_{v=1}^{k} \Delta'_{q,v} = \sum_{r=1}^{m} \Delta_{qr}$. Thus Δ' is a correspondence matrix. Moreover,
   - If q is such that $M_{pq} \ne 0$, then $\sum_{i=1}^{k} \Delta'_{q,i} = \sum_{r=1}^{m} \Delta_{q,r} = 1$;

- For all $1 \leq i \leq k$, $[M_p\Delta']_i = \sum_{r\in\mathcal{N}(i)}([M_p\Delta]_r)$. As a consequence, $\varphi(u)(M_p\Delta) = \varphi'(j)(M_p\Delta')$ holds.
- If q, v are such that $\Delta'_{q,v} \neq 0$, then there exists $r \in \mathcal{N}(v)$ such that $\Delta_{q,r} \neq 0$, thus $q\mathcal{R}'v$.

Finally, $\mathcal{R}'$ is a satisfaction relation. By construction $o_P\mathcal{R}'o$, thus it holds that $P \models S$. □

It is easy to see that normalization preserves determinism.

4. Refinement

Comparing specifications is central to stepwise design methodologies. Systematic comparison enables simplification of specifications (abstraction) and adding details to specifications (elaboration). Usually specifications are compared using a *refinement* relation. Roughly, if S_1 refines S_2, then any model of S_1 is also a model of S_2.

We will now introduce two notions of refinement for CMCs that extend two well known refinements for IMCs [11,16]. We not only generalize these refinements, but, unlike [11,16], we also characterize them in terms of implementation set inclusion – also called *thorough refinement* – and computational complexity. We start with the definition of refinements, then we propose algorithms to compute them.

4.1. Refinement relations

The strong refinement between IMCs, by Jonsson and Larsen [11], extends to CMCs in the following way:

Definition 9 (*Strong Refinement*). Let $S_1 = \langle\{1,\ldots,k_1\}, o_1, \varphi_1, A_1, V_1\rangle$ and $S_2 = \langle\{1,\ldots,k_2\}, o_2, \varphi_2, A_2, V_2\rangle$ be CMCs with $A_2 \subseteq A_1$. A relation $\mathcal{R} \subseteq \{1,\ldots,k_1\} \times \{1,\ldots,k_2\}$ is a *strong refinement relation* between states of S_1 and S_2 iff whenever $v\mathcal{R}u$, then

1. $V_1(v)\downarrow_{A_2} \subseteq V_2(u)$, and
2. there exists a correspondence matrix $\Delta \in [0,1]^{k_1\times k_2}$ such that for all probability distribution vectors $x \in [0,1]^{k_1}$ if $\varphi_1(v)(x)$ holds, then
   - for all $1 \leq i \leq k_1, x_i \neq 0 \implies \sum_{j=1}^{k_2} \Delta_{ij} = 1$;
   - $\varphi_2(u)(x\Delta)$ holds and
   - if $\Delta_{v'u'} \neq 0$, then $v'\mathcal{R}u'$.

We say that S_1 strongly refines S_2, written $S_1 \preceq_S S_2$, iff $o_1\mathcal{R}o_2$.

Strong refinement imposes a "fixed-in-advance" correspondence matrix Δ regardless of the probability distribution satisfying the constraint function. In contrast, the *weak refinement*, which generalizes the one proposed in [16] for IMCs, allows choosing a different correspondence matrix for each probability distribution satisfying the constraint:

Definition 10 (*Weak Refinement*). Let $S_1 = \langle\{1,\ldots,k_1\}, o_1, \varphi_1, A_1, V_1\rangle$ and $S_2 = \langle\{1,\ldots,k_2\}, o_2, \varphi_2, A_2, V_2\rangle$ be CMCs with $A_2 \subseteq A_1$. The relation $\mathcal{R} \subseteq \{1,\ldots,k_1\} \times \{1,\ldots,k_2\}$ is a *weak refinement relation* iff whenever $v\mathcal{R}u$, then:

1. $V_1(v)\downarrow_{A_2} \subseteq V_2(u)$ and
2. for any distribution $x \in [0,1]^{k_1}$ satisfying $\varphi_1(v)(x)$, there exists a matrix $\Delta \in [0,1]^{k_1\times k_2}$ such that
   - for all $1 \leq i \leq k_1, x_i \neq 0 \implies \sum_{j=1}^{k_2} \Delta_{ij} = 1$;
   - $\varphi_2(u)(x\Delta)$ holds and
   - if $\Delta_{v'u'} \neq 0$, then $v'\mathcal{R}u'$.

CMC S_1 (weakly) refines S_2, written $S_1 \preceq S_2$, iff $o_1\mathcal{R}o_2$.

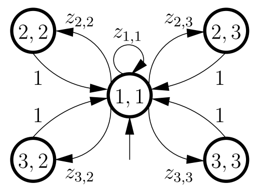
Example. Fig. 8c illustrates a family of correspondence matrices parametrized by γ, witnessing the weak refinement between initial states of S_3 and S_4 (defined in Fig. 8a–b). The actual matrix used in proving the weak refinement depends on the probability distribution vector z that satisfies the constraint function φ_3 of state $(1,1)$. Take $\gamma = \frac{0.7-z_{2,2}}{z_{2,3}}$ if $z_{2,2} \leq 0.7$ and $\gamma = \frac{0.8-z_{2,2}}{z_{2,3}}$ otherwise. It is easy to see that $\varphi_3((1,1))(z)$ implies $\varphi_4(1)(z\Delta)$.

Clearly, the existence a strong refinement relation implies the existence of a weak refinement relation, as every strong refinement is a also a weak refinement. Furthermore, both weak and strong refinements imply implementation set inclusion, as shown by the following theorem:

Theorem 11. *Let $S_1 = \langle\{1,\ldots,k_1\}, o_1, \varphi_1, A_1, V_1\rangle$ and $S_2 = \langle\{1,\ldots,k_2\}, o_2, \varphi_2, A_2, V_2\rangle$ be two CMCs. Assume $S_1 \preceq S_2$, we prove that $[\![S_1]\!] \subseteq [\![S_2]\!]$.*

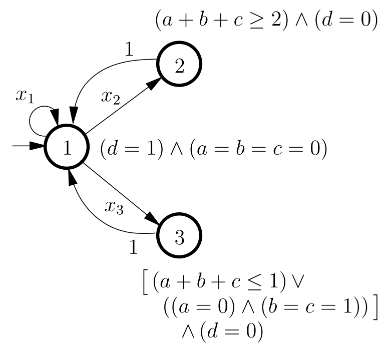
Proof. Since $S_1 \preceq S_2$, there exists a weak refinement relation $\mathcal{R} \subseteq \{1,\ldots,k_1\} \times \{1,\ldots,k_2\}$ such that $o_1\mathcal{R}o_2$. Consider $P = \langle\{1,\ldots n\}, o_P, M, A_P, V_P\rangle$ such that $P \models S_1$. By definition there exists a satisfaction relation $\mathcal{R}' \subseteq \{1,\ldots,n\} \times \{1,\ldots,k_1\}$ such that $o_P\mathcal{R}'o_1$.

Consider the relation $\mathcal{R}'' \subseteq \{1,\ldots,n\} \times \{1,\ldots,k_2\}$, such that $p\mathcal{R}''u$ iff. there exists $v \in \{1,\ldots,k_1\}$ such that $p\mathcal{R}'v$ and $v\mathcal{R}u$. We prove that $\mathcal{R}''$ is a satisfaction relation. First, it is clear that $A_2 \subseteq A_1 \subseteq A_P$. Now, consider p, u such that $p\mathcal{R}''u$, so there exists v such that $p\mathcal{R}'v$ and $v\mathcal{R}u$. Since $V_P(p)\downarrow_{A_1} \in V_1(v)$ and $V_1(v)\downarrow_{A_2} \subseteq V_2(u)$, we have that $V_P(p)\downarrow_{A_2} \in V_2(u)$.



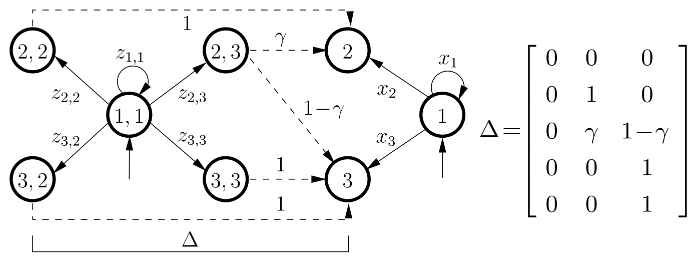
$$\begin{aligned}\varphi_3(1,1)(Z) \equiv{} & (\forall j.\ z_{1,j} = 0)\\ \wedge\ & (z_{2,2} + z_{2,3} \geq 0.7)\\ \wedge\ & (z_{2,2}+z_{2,3}+z_{3,2}+z_{3,3} = 1)\\ \wedge\ & (z_{2,3} + z_{3,3} \geq 0.2)\end{aligned}$$

(a) $S_3 = S_1 \wedge S_2$. Constraints on propositions, pairwise conjunctions of constraints of S_1 and S_2, are left out to avoid clutter.


$(a + b + c \geq 2) \wedge (d = 0)$
1
2
x_1
x_2
1
$(d = 1) \wedge (a = b = c = 0)$
x_3
3
1
$\big[(a + b + c \leq 1) \vee ((a = 0) \wedge (b = c = 1))\big] \wedge (d = 0)$

$$\begin{aligned}\varphi_4(1)(x) \equiv{} & (x_2 \geq 0.7)\\ \wedge\, & (x_3 \geq 0.2) \wedge (x_2 + x_3 = 1)\end{aligned}$$

(b) CMC S_4 generalizing S_3, so $S_3 \preceq S_4$.

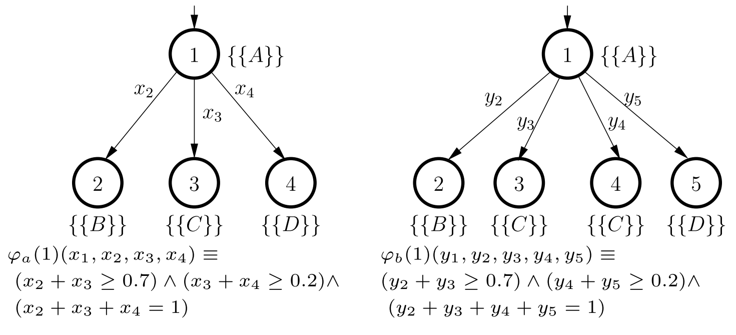

1
2, 2
$z_{1,1}$
2, 3
γ
2
x_1
$z_{2,2}$
$z_{2,3}$
$1-\gamma$
x_2
1, 1
1
$z_{3,2}$
$z_{3,3}$
x_3
3, 2
3, 3
1
3
1
Δ
$\Delta = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & \gamma & 1-\gamma \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}$

(c) Weak refinement for initial states of S_3 and S_4.

Fig. 8. Examples of refinement and conjunction.

We now build a correspondence matrix Δ''. Consider the pth row of M, $M_p \in [0,1]^n$. Let $\Delta' \in [0,1]^{n\times k_1}$ be a correspondence matrix witnessing $p\mathcal{R}'v$. Let $y = M_p\Delta' \in [0,1]^{k_1}$. By Definition 4 we have $\varphi_1(v)(y)$. Let $\Delta \in [0,1]^{k_1\times k_2}$ be the correspondence matrix witnessing $v\mathcal{R}u$ and define $\Delta'' = \Delta'\Delta \in [0,1]^{n\times k_2}$. By Lemma 1, Δ'' is also a correspondence matrix. We prove that Δ'' satisfies the axioms of Definition 4.

1. Let $1 \leq p' \leq n$ such that $M_{pp'} \neq 0$. As a consequence, $\sum_{q=1}^{k_1} \Delta'_{p'q} = 1$. We want to prove that $\sum_{j=1}^{k_2} \Delta''_{p'j} = 1$.
$$\sum_{j=1}^{k_2} \Delta''_{p'j} = \sum_{j=1}^{k_2}\left(\sum_{q=1}^{k_1} \Delta'_{p'q}\Delta_{qj}\right) = \sum_{q=1}^{k_1}\left(\Delta'_{p'q}\sum_{j=1}^{k_2}\Delta_{qj}\right).$$
Let q such that $\Delta'_{p'q} \neq 0$. It is then clear that $y_q \geq M_{pp'}\Delta'_{p'q} > 0$. As Δ is a witness of $v\mathcal{R}u$, we have, by the definition of weak refinement, $\sum_{j=1}^{k_2} \Delta_{qj} = 1$. Finally, this implies that $\sum_{j=1}^{k_2} \Delta''_{p'j} = 1$.
2. By Definition 10, since $\varphi_1(v)(M_p\Delta)$ holds, then $\varphi_2(u)(M_p\Delta'')$ holds.


1
{{A}}
x_2
x_3
x_4
2
3
4
{{B}}
{{C}}
{{D}}
$\varphi_a(1)(x_1, x_2, x_3, x_4) \equiv$
$(x_2 + x_3 \geq 0.7) \wedge (x_3 + x_4 \geq 0.2) \wedge$
$(x_2 + x_3 + x_4 = 1)$
1
{{A}}
y_2
y_3
y_4
y_5
2
3
4
5
{{B}}
{{C}}
{{C}}
{{D}}
$\varphi_b(1)(y_1, y_2, y_3, y_4, y_5) \equiv$
$(y_2 + y_3 \geq 0.7) \wedge (y_4 + y_5 \geq 0.2) \wedge$
$(y_2 + y_3 + y_4 + y_5 = 1)$

(a) CMC S_a. (b) CMC S_b.

Fig. 9. Distinguishing weak and strong refinement. S_a weakly, but not strongly, refines S_b. Here, and elsewhere, self-targeting loop transitions with probability 1 have been elided (in states with no outgoing arrows).

3. Let p', u' such that $\Delta''_{p'u'} \neq 0$. By construction, it is clear that there exists v' such that $\Delta'_{p'v'} \neq 0$ and $\Delta_{v'u'} \neq 0$. By definition of Δ' and Δ, this implies that $p'\mathcal{R}'v'$ and $v'\mathcal{R}u'$, thus $p'\mathcal{R}''u'$.

From 1–3, we can conclude that $\mathcal{R}''$ is a satisfaction relation. Since $o_P\mathcal{R}''o_2$, we have $P \in [\![S_2]\!]$ and $[\![S_1]\!] \subseteq [\![S_2]\!]$. □

In Section 8, we shall see that the converse holds for a particular class of CMCs. However, this is not the case in general: strong refinement is strictly stronger than weak refinement, which is strictly stronger than implementation set inclusion. Formally, we have the following proposition.

Proposition 12. *There exist CMCs S_a, S_b, S_c and S_d such that*

- *S_a weakly refines S_b, and S_a does not strongly refine S_b;*
- *$[\![S_c]\!] \subseteq [\![S_d]\!]$, and S_c does not weakly refine S_d.*

Proof. • Consider the CMCs S_a and S_b given in Fig. 9a and b, respectively. By x_a (resp. y_b) we denote state x in S_a (resp. y in S_b). We first show that there exists a weak refinement relation $\mathcal{R}$ such that $S_a \preceq S_b$, with $1_a\mathcal{R}1_b$. We then show that there exists no strong refinement relation between S_a and S_b.

1. Let $\mathcal{R} = \{(1_a, 1_b), (2_a, 2_b), (3_a, 3_b), (3_a, 4_b), (4_a, 5_b)\}$. We show that $\mathcal{R}$ is a weak refinement relation. We first focus on building the correspondence matrix for the pair $(1_a, 1_b)$. Let x be a distribution satisfying the constraint in 1_a. Let $\gamma = \frac{0.7-x_2}{x_3}$ if $x_2 \leq 0.7$ and $\frac{0.8-x_2}{x_3}$ otherwise. As x satisfies $\varphi_a(1_a)$, we have $0 \leq \gamma \leq 1$. Consider the correspondence matrix Δ_x given in Fig. 10

   It is easy to see that for all valuations x satisfying $\varphi_a(1_a)$, $\varphi_b(1_b)(x\Delta_x)$ also holds. The correspondence matrices for the other pairs in $\mathcal{R}$ are trivial. Thus $\mathcal{R}$ is a weak refinement relation between S_a and S_b.
2. Suppose that there exists a strong refinement relation $\mathcal{R}'$ such that $1_a\mathcal{R}'1_b$. Let Δ be the correspondence matrix witnessing $1_a\mathcal{R}'1_b$. Since 2_a, 3_a and 4_a can all be reached from 1_a with an admissible transition, the sum of the elements in the corresponding rows in Δ must be one. From the valuations of the states, we obtain that Δ is of the type given in Fig. 10, with $0 \leq a \leq 1$.

   Moreover, if $\mathcal{R}'$ is a strong refinement relation, then we have that for all valuation x satisfying $\varphi_a(1_a)$, $\varphi_b(1_b)(x\Delta)$ also holds.

   Let $x^1 = (0, 0.6, 0.1, 0.3)$ and $x^2 = (0, 0.8, 0.1, 0.1)$. Both x^1 and x^2 satisfy $\varphi_a(1_a)$. If there exists a strong refinement, this implies that $\varphi_b(1_b)(x^1\Delta)$ and $\varphi_b(1_b)(x^2\Delta)$ also hold. However, $\varphi_b(1_b)(x^1\Delta) = 1$ implies that $a \geq 1$ and $\varphi_b(1_b)(x^2\Delta)$ implies that $a \leq 0$.

   It is thus impossible to find a unique correspondence matrix working for all the "valid" valuations of the outgoing transitions of 1_a. As a consequence, there cannot exist a strong refinement relation $\mathcal{R}'$ such that $1_a\mathcal{R}'1_b$.

• Consider the CMCs S_c and S_d given in Fig. 11a and b. It is easy to see that S_c and S_d share the same set of implementations. However, due to the constraints, state 2 of S_c cannot refine any state of S_d. As a consequence, S_c cannot refine S_d. □

So our refinement relations for CMCs can be ordered from finest to coarsest: the strong refinement, the weak refinement, and the implementation set inclusion. As the implementation set inclusion is the *ultimate* refinement, checking finer refinements is used as a pragmatic syntax-driven, but sound, way of deciding it.

As we shall see in the next section, the algorithms for checking weak and strong refinements are polynomial in the number of states, but the treatment of each state depends on the complexity of the constraints. For the case of implementation set inclusion, the algorithm is exponential in the number of states. Checking implementation set inclusion seems thus harder than checking weak or strong refinement. In Section 8, we will propose a class of CMCs for which strong and weak refinements coincide with implementation set inclusion.

$$\Delta_x = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & \gamma & (1-\gamma) & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \quad \Delta = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & a & (1-a) & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Fig. 10. Correspondence matrices for $S_a \preceq S_b$.

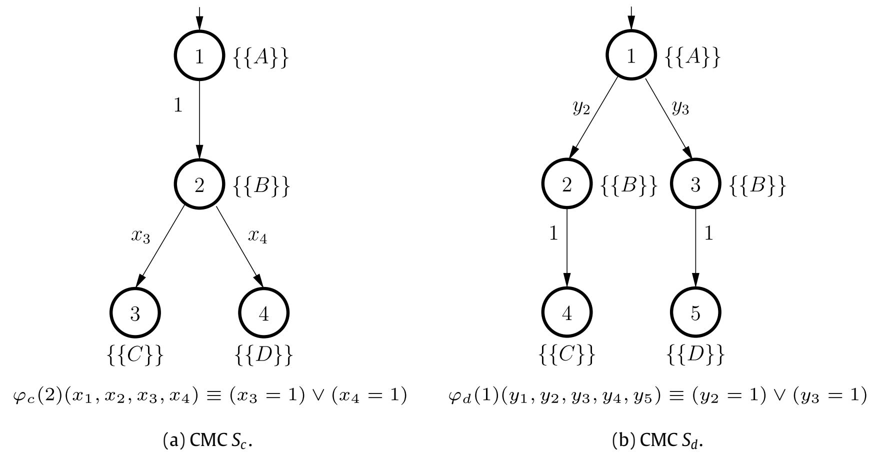


Fig. 11. Weak refinement versus model inclusion: $[\![S_c]\!] \subseteq [\![S_d]\!]$ but $S_c \not\preceq S_d$.

4.2. Algorithms for computing refinements

We now discuss algorithms for checking implementation set inclusion and refinements.

We start with algorithms for checking weak and strong refinements between two CMCs $S_1 = \langle\{1, \ldots, k_1\}, o_1, \varphi_1, A_1, V_1\rangle$ and $S_2 = \langle\{1, \ldots, k_2\}, o_2, \varphi_2, A_2, V_2\rangle$ with $k_1, k_2 \leq n$. Checking whether a relation $\mathcal{R} \subseteq \{1, \ldots, k_1\} \times \{1, \ldots, k_2\}$ is a strong (resp. weak) refinement relation reduces to checking, for all $(i, j) \in \mathcal{R}$, the validity of the following *refinement formulas*: $\exists\Delta, \forall x, \varphi_1(i)(x) \Rightarrow \varphi_2(j)(x\Delta) \wedge \bigwedge_{i'}(\sum_{j'} \Delta_{i'j'} = 1) \wedge \bigwedge_{i',j'}(i'\mathcal{R}j' \vee \Delta_{i'j'} = 0)$ for the strong refinement, and $\forall x, \varphi_1(i)(x) \Rightarrow \exists\Delta, \varphi_2(j)(x\Delta) \wedge \bigwedge_{i'}(\sum_{j'} \Delta_{i'j'} = 1) \wedge \bigwedge_{i',j'}(i'\mathcal{R}j' \vee \Delta_{i'j'} = 0)$ for the weak refinement. Strong and weak refinements can be decided by iterated strengthening of $\mathcal{R}$ with refinement formulas, starting from $\mathcal{R}_0 = \{(i, j)|V_1(i) \downarrow_{A_2} \subseteq V_2(j)\}$, until either $(o_1, o_2) \notin \mathcal{R}$, in which case S_1 does not strongly (resp. weakly) refine S_2, or $\mathcal{R}$ is found to be a strong (resp. weak) refinement.

The exact complexity of the algorithm depends on the type of constraints that are used in the specifications. As an example, consider that all the constraints in S_1 and S_2 are polynomials of degree d with less than k bound variables — we shall see that polynomial constraints is the smallest class under which CMCs are closed. There, deciding refinement formulas can be done by quantifier elimination. When the number of quantifier alternations is constant, the *cylindrical algebraic decomposition algorithm* [23,24], implemented in Maple [25], performs this quantifier elimination in time double exponential in the number of variables. Consequently, refinement can be checked in $O(n^2 2^{2^{n^2}})$ time.

However, considering constraints φ which contain only existential quantifiers, quantifier alternation is either one or two for strong refinement and exactly one for weak refinement. There are quantifier elimination algorithms that have a worst case complexity of a single exponential only in the number of variables, although they are double exponential in the number of quantifier alternations [26]. Thanks to these algorithms, deciding whether $\mathcal{R}$ is a strong (resp. weak) refinement relation can be done in time single exponential in the number of states n and k, the number of bound variables appearing in the constraints: $O(n^2 s^{P(n,k)} d^{P(n,k)})$, where P is a polynomial.

We now turn to the case of implementation set inclusion. In [11], Larsen and Jonsson proposed an algorithm for solving this problem for the case of IMCs. This algorithm directly extends to CMCs. The main difference with the algorithms for solving weak and strong refinements is that the algorithm for implementation set inclusion is exponential in the number of states.

Finally, let us mention that lower bounds for the strong and weak refinement checking remain open problems. On the other hand, in [27], we have shown that implementation set inclusion is EXPTIME-hard for IMCs, hence providing a lower bound also for CMCs.

5. Conjunction

Conjunction combines requirements of several specifications.

Definition 13 (*Conjunction*). Let $S_1 = \langle\{1,\ldots,k_1\}, o_1, \varphi_1, A_1, V_1\rangle$ and $S_2 = \langle\{1,\ldots,k_2\}, o_2, \varphi_2, A_2, V_2\rangle$ be two CMCs. The conjunction of S_1 and S_2, written $S_1 \wedge S_2$, is the CMC $S = \langle\{1,\ldots,k_1\}\times\{1,\ldots,k_2\}, (o_1,o_2), \varphi, A, V\rangle$ with $A = A_1 \cup A_2$, $V((u,v)) = V_1(u){\uparrow^A} \cap V_2(v){\uparrow^A}$, and

$$\varphi((u,v))(x_{1,1},x_{1,2},\ldots,x_{2,1},\ldots,x_{k_1,k_2}) \equiv \ \varphi_1(u)\left(\sum_{j=1}^{k_2} x_{1,j},\ldots,\sum_{j=1}^{k_2} x_{k_1,j}\right) \wedge \varphi_2(v)\left(\sum_{i=1}^{k_1} x_{i,1},\ldots,\sum_{i=1}^{k_1} x_{i,k_2}\right).$$

Conjunction may introduce inconsistent states. Indeed, the intersection between the sets of valuations of two states may be empty (see state $(2,3)$ in Fig. 1). Conjunction should thus normally be followed by applying the pruning operator β^*. As already stated in the Introduction, the result of conjoining two IMCs is not an IMC in general, but a CMC whose constraint functions are systems of linear inequalities. Fig. 8a depicts a CMC S_3 expressing the conjunction of IMCs S_1 and S_2 (see Fig. 3a–b). The constraint $z_{2,3} + z_{3,3} \geq 0.2$ in state $(1,1)$ cannot be expressed as an interval.

As expected, conjunction of two specifications coincides with their greatest lower bound with respect to the weak refinement (also called *shared refinement*).

Theorem 14. *Let S_1, S_2 and S_3 be three CMCs. We have* (a) $((S_1 \wedge S_2) \preceq S_1)$ *and* $((S_1 \wedge S_2) \preceq S_2)$ *and* (b) *if* $(S_3 \preceq S_1)$ *and* $(S_3 \preceq S_2)$, *then* $S_3 \preceq (S_1 \wedge S_2)$.

Proof. We separately prove the two items of the theorem. Let $\{1,\ldots,k_1\}$, $\{1,\ldots,k_2\}$, and $\{1,\ldots,k_3\}$ be the sets of states of S_1, S_2, and S_3, respectively.
(a) Let $S_1 \wedge S_2 = S = \langle\{1,\ldots,k_1\}\times\{1,\ldots,k_2\}, o, \varphi, A, V\rangle$. Let $\mathcal{R} \subseteq (\{1,\ldots,k_1\}\times\{1,\ldots,k_2\}) \times \{1,\ldots,k_1\}$ such that $(u,v)\mathcal{R}w \iff u = w$. We will prove that $\mathcal{R}$ is a strong refinement relation. Let $u \in \{1,\ldots,k_1\}$ and $v \in \{1,\ldots,k_2\}$. We have $(u,v)\mathcal{R}u$.

1. By definition of S obtain $V((u,v)){\downarrow_{A_1}} = (V_1(u){\uparrow^A} \cap V_2(v){\uparrow^A}){\downarrow_{A_1}} \subseteq V_1(u)$.
2. Let $\Delta \in [0,1]^{(k_1k_2)\times k_1}$ such that $\Delta_{(i,j),i} = 1$ and $\Delta_{(i,j),k} = 0$ if $k \neq i$. We now prove that it satisfies the axioms of a satisfaction relation for $(u,v)\mathcal{R}u$ stated in Definition 4:
   - Then we have $\forall (i,j).\ \sum_{k=1}^{k_1} \Delta_{(i,j),k} = 1$.
   - If $x \in [0,1]^{(k_1k_2)}$ is such that $\varphi((u,v))(x)$, it implies by definition that $\varphi_1(u)(\sum_{j=1}^{k_2} x_{1,j},\ldots,\sum_{j=1}^{k_2} x_{k_1,j}) = \varphi_1(u)(x\Delta)$ holds.
   - If $\Delta_{(u',v'),w'} \neq 0$, we have by definition $u' = w'$ and $(u',v')\mathcal{R}u'$.

We conclude that $\mathcal{R}$ is a strong, and thus also a weak, refinement relation. Since $(o_1,o_2)\mathcal{R}o_1$, we have $S_1 \wedge S_2 \preceq S_1$. By symmetry, we also have $S_1 \wedge S_2 \preceq S_2$.
(b) Assume $S_3 \preceq S_1$ and $S_3 \preceq S_2$. By definition, there exist refinement relations $\mathcal{R}_1 \subseteq \{1,\ldots,k_3\}\times\{1,\ldots,k_1\}$ and $\mathcal{R}_2 \subseteq \{1,\ldots,k_3\}\times\{1,\ldots,k_2\}$ such that $o_3\mathcal{R}_1o_1$ and $o_3\mathcal{R}_2o_2$. Let $S_1 \wedge S_2 = S = \langle\{1,\ldots,k_1\}\times\{1,\ldots,k_2\}, o, \varphi, A, V\rangle$.

Let $\mathcal{R} \subseteq \{1,\ldots,k_3\}\times(\{1,\ldots,k_1\}\times\{1,\ldots,k_2\})$ such that $u\mathcal{R}(v,w) \iff u\mathcal{R}_1v$ and $u\mathcal{R}_2w$. We now prove that $\mathcal{R}$ is a weak refinement relation.

Consider u, v, w such that $u\mathcal{R}(v,w)$.

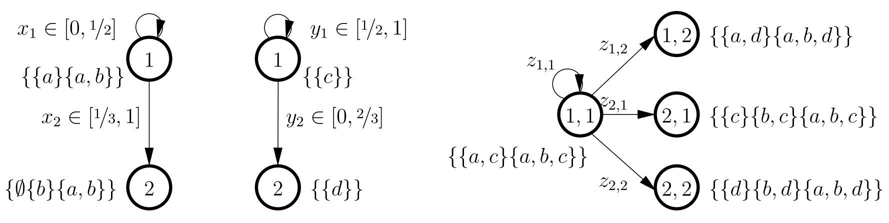
1. By definition, we have $V_3(u){\downarrow_{A_1}} \subseteq V_1(v)$ and $V_3(u){\downarrow_{A_2}} \subseteq V_2(w)$. As a consequence, $V_3(u){\downarrow_A} \subseteq V((v,w))$.
2. Let $x \in [0,1]^{k_3}$ such that $\varphi_3(u)(x)$. Consider the correspondence matrices $\Delta \in [0,1]^{k_3\times k_1}$ and $\Delta' \in [0,1]^{k_3\times k_2}$ given by $u\mathcal{R}_1v$ and $u\mathcal{R}_2w$ for the transition vector x. Let $\Delta'' \in [0,1]^{k_3\times(k_1k_2)}$ be a new matrix such that $\Delta'' = \Delta \otimes \Delta'$. By Lemma 1, Δ'' is a correspondence matrix. We now prove that it satisfies the axioms of a refinement relation for $u\mathcal{R}(v,w)$:
   - Let $1 \leq i \leq k_3$ such that $x_i \neq 0$. By definition of Δ and Δ', we have $\sum_{j=1}^{k_1} \Delta_{ij} = 1$ and $\sum_{q=1}^{k_2} \Delta'_{iq} = 1$, so $\sum_{(j,q)\in\{1,\ldots,k_1\}\times\{1,\ldots,k_2\}} \Delta''_{i(j,q)} = (\sum_{j=1}^{k_1} \Delta_{ij})(\sum_{q=1}^{k_2} \Delta'_{iq}) = 1$.
   - By definitions of Δ and Δ', both $\varphi_1(v)(x\Delta)$ and $\varphi_2(w)(x\Delta')$ hold. Let $x' = x\Delta''$. It is clear that $x\Delta = (\sum_{j=1}^{k_2} x'_{(1,j)},\ldots,\sum_{j=1}^{k_2} x'_{(k_1,j)})$ and $x\Delta' = (\sum_{i=1}^{k_1} x'_{(i,1)},\ldots,\sum_{i=1}^{k_1} x'_{(i,k_2)})$. As a consequence, $\varphi((v,w))(x\Delta'')$ holds.
   - Let u', v', w' such that $\Delta''_{u'(v',w')} \neq 0$. By construction, this implies $\Delta_{u'v'} \neq 0$ and $\Delta'_{u'w'} \neq 0$. As a consequence, $u'\mathcal{R}_1v'$ and $u'\mathcal{R}_2w'$, thus $u'\mathcal{R}(v',w')$.

We conclude that $\mathcal{R}$ is a weak refinement relation. Since $o_3\mathcal{R}(o_1,o_2)$, we have $S_3 \preceq (S_1 \wedge S_2)$. □

The first consequence of the above theorem is that conjunction with another specification is a monotonic operator with respect to weak refinement. Furthermore, as it follows from the later results of Section 8, the set of implementations of a conjunction of two *deterministic* specifications S_1 and S_2 coincides with the intersection of implementation sets of S_1 and S_2 (the greatest lower bound in the lattice of implementation sets).

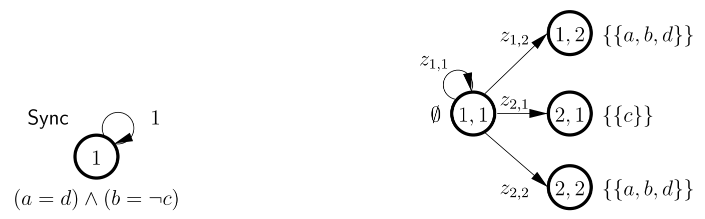
6. Separation of concerns in parallel composition of specifications

Let us now turn to parallel composition. We first remark, that in concurrency theory it is customary to combine parallel composition with synchronization—for example in many process algebras the same rules are used to explain how parallel processes evolve, and how they communicate. In this work we take a different approach, separating these two



(a) Two CMCs S and S'.

(b) $S \parallel S'$.


Sync
1
$(a = d) \wedge (b = \neg c)$
$z_{1,1}$ $z_{1,2}$ $z_{2,1}$ $z_{2,2}$
$\emptyset$
$\{\{a,b,d\}\}$
$\{\{c\}\}$
$\{\{a,b,d\}\}$

(c) Synchronizer Sync.

(d) $(S \parallel S') \wedge$ Sync.

Fig. 12. Parallel composition and synchronization of CMCs.

concerns— parallel composition from synchronization. The choices regarding sets of valuations and the stochastic choices are independent from each other.

First, components are composed into a kind of product—effectively just a vector of stochastically independent entities. Second, the product is synchronized on valuations by constraining its behavior, to model handshake communication. For example, assume that one component should be in a state where b holds, whenever $\overline{b}$ holds in another one. An independent product of these two components will likely contain states where only one of the propositions is present, but not the other. Then a synchronization operator is applied that will eliminate these states from the product, ensuring that the two propositions b and $\overline{b}$ always co-occur.

This design has two significant advantages. First, it allows modeling very diverse synchronization mechanisms. For CMCs synchronization goes far beyond matching label names, like in the above example. The synchronization is specified as a part of the model, where it can express complex propositional constraints between propositions (for example a safety property). It can be stateful and probabilistic.

Second, as we will see, we will obtain synchronization by simply using conjunction. This very elegantly exploits the prior results on conjunction, as the synchronization operator turns out to be realizable using conjunction.

Remark 1. The principle of separation of concerns is intensively used in the definition of parallel composition for many systems that mix stochastic and non-deterministic choices, among them many theories for probabilistic process algebra [17,19]. Similar principles also apply for continuous-time stochastic models, in a slightly different setting based on CTMCs [20]. In Section 10, in order to argue that such a design is reasonable, we will show that our parallel composition covers the one of probabilistic automata [17]—which is a widely accepted and appreciated operator.

We start by showing how systems and specifications are composed in a nonsynchronizing manner, then we introduce synchronization. The non-synchronizing *independent* parallel composition is largely just a product of two MCs (or CMCs):

Definition 15 (*Parallel Composition of MCs*). Let $P_1 = \langle \{1, \dots, n_1\}, o_1, M', A_1, V_1 \rangle$ and $P_2 = \langle \{1, \dots, n_2\}, o_2, M'', A_2, V_2 \rangle$ be two MCs with $A_1 \cap A_2 = \emptyset$. The parallel composition of P_1 and P_2 is the MC $P_1 \parallel P_2 = \langle \{1, \dots, n_1\} \times \{1, \dots, n_2\}, (o_1, o_2), M, A_1 \cup A_2, V \rangle$, where: $M \in [0, 1]^{(n_1 n_2) \times (n_1 n_2)}$ is such that $M = M' \odot M''$ and $V((p, q)) = V_1(p) \cup V_2(q)$.

For CMCs we have the following definition.

Definition 16 (*Parallel Composition of CMCs*). Let $S_1 = \langle \{1, \dots, k_1\}, o_1, \varphi_1, A_1, V_1 \rangle$ and $S_2 = \langle \{1, \dots, k_2\}, o_2, \varphi_2, A_2, V_2 \rangle$ be CMCs with $A_1 \cap A_2 = \emptyset$. The parallel composition of S_1 and S_2 is the CMC $S_1 \parallel S_2 = \langle \{1, \dots, k_1\} \times \{1, \dots, k_2\}, (o_1, o_2), \varphi, A_1 \cup A_2, V \rangle$, where

$$\varphi((u, v))(z_{1,1}, z_{1,2}, \dots, z_{2,1}, \dots, z_{k_1,k_2}) \equiv \exists x_1, \dots, x_{k_1}, y_1, \dots, y_{k_2} \in [0, 1]. \forall (i, j) \in \{1, \dots, k_1\} \times \{1, \dots, k_2\}.$$
$$z_{i,j} = x_i y_j \quad \text{and} \quad \varphi_1(u)(x_1, \dots, x_{k_1}) = \varphi_2(v)(y_1, \dots, y_{k_2}) = 1.$$

Finally, $V((u, v)) = \{Q_1 \cup Q_2 \mid Q_1 \in V_1(u), Q_2 \in V_2(v)\}$.

In the Introduction we have demonstrated that IMCs are not closed under conjunction. Here, it is worth mentioning that IMCs are not closed under parallel composition, even under the independent one. Consider IMCs S and S' given in Fig. 12a and their parallel composition $S \parallel S'$ given in Fig. 12b. Assume first that $S \parallel S'$ is an IMC. As a variable $z_{i,j}$ is the product of two variables x_i and y_j, if $S \parallel S'$ is an IMC, then one can show that the interval for $z_{i,j}$ is obtained by computing the products of the bounds of the intervals over which x_i and y_j range. Hence, we can show that $z_{1,1} \in [0, 1/2], z_{1,2} \in [0, 1/3], z_{2,1} \in [1/6, 1], z_{2,2} \in [0, 2/3]$. Let $[a, b]$ be the interval for the constraint $z_{i,j}$, it is easy to see that there exist implementations I_1 of S_1 and I_2 of S_2 such that $I_1 \parallel I_2$ satisfies the constraint $z_{i,j} = a$ (resp. $z_{i,j} = b$). However, while each bound of each interval can be satisfied independently, some points in the polytope defined by the intervals and the constraint $\sum z_{i,j} = 1$ cannot be reached. As an example, consider $z_{1,1} = 0, z_{1,2} = 1/3, z_{2,1} = 1/3, z_{2,2} = 1/3$. It is clearly inside the polytope, but one cannot find an implementation I of $S \parallel S'$ satisfying the constraints given by the parallel composition. Indeed, having $z_{1,1} = 0$ implies that $x_1 = 0$ and thus that $z_{1,2} = 0$.

Theorem 17. *If S'_1, S'_2, S_1, S_2 are CMCs, then $S'_1 \preceq S_1$ and $S'_2 \preceq S_2$ implies $S'_1 \parallel S'_2 \preceq S_1 \parallel S_2$, so the weak refinement is a precongruence with respect to parallel composition. Consequently, for any MCs P_1 and P_2 we have that $P_1 \models S_1 \wedge P_2 \models S_2$ implies $P_1 \parallel P_2 \models S_1 \parallel S_2$.*

Proof. Let

$$\begin{aligned}
S_1 &= \langle\{1,\dots,k_1\}, o_1, \varphi_1, A_1, V_1\rangle,\\
S_2 &= \langle\{1,\dots,k_2\}, o_2, \varphi_2, A_2, V_2\rangle,\\
S'_1 &= \langle\{1,\dots,k'_1\}, o'_1, \varphi'_1, A'_1, V'_1\rangle,\\
S'_2 &= \langle\{1,\dots,k'_2\}, o'_2, \varphi'_2, A'_2, V'_2\rangle,\\
S\ &= \langle\{1,\dots,k_1\}\times\{1,\dots,k_2\}, (o_1,o_2), \varphi, A, V\rangle = S_1 \parallel S_2,\\
S'\ &= \langle\{1,\dots,k'_1\}\times\{1,\dots,k'_2\}, (o'_1,o'_2), \varphi', A', V'\rangle = S'_1 \parallel S'_2,
\end{aligned}$$

be CMCs with $A = A_1 \cup A_2$ and $A' = A'_1 \cup A'_2$. Assume that $S'_1 \preceq S_1$ and $S'_2 \preceq S_2$.

By definition, there exist two weak refinement relations $\mathcal{R}_1$ and $\mathcal{R}_2$ such that $o'_1\mathcal{R}_1 o_1$ and $o'_2\mathcal{R}_2 o_2$. Define $\mathcal{R}$ such that $(u', v')\mathcal{R}(u, v) \iff u'\mathcal{R}_1 u$ and $v'\mathcal{R}_2 v$. Consider now such (u', v') and (u, v). We prove that $\mathcal{R}$ satisfies the axioms of a weak refinement relation between (u', v') and (u, v):

1. Note that from $u'\mathcal{R}_1 u$ (resp. $v'\mathcal{R}_2 v$) it follows that $A'_1 \subseteq A_1$ (resp. $A'_2 \subseteq A'_2$) and further $A'_1 \cap A_2 = A_1 \cap A'_2 = \emptyset$. We have:

$$\begin{aligned}
V'((u',v'))\downarrow_A &= \left\{(Q_1 \cup Q_2)\downarrow_{A_1\cup A_2} \mid Q_1 \in V'_1(u'), Q_2 \in V'_2(v')\right\}\\
&= \{Q_1\downarrow_{A_1} \cup Q_2\downarrow_{A_2} \mid Q_1 \in V'_1(u'), Q_2 \in V'_2(v')\} \subseteq \{Q_1 \cup Q_2 \mid Q_1 \in V_1(u), Q_2 \in V_2(v)\} = V((u,v)).
\end{aligned}$$

2. Let $z' \in [0,1]^{(k'_1k'_2)}$ such that $\varphi'(u',v')(z')$. We now build the correspondence matrix Δ witnessing $(u',v')\mathcal{R}(u,v)$. Consider the correspondence matrices $\Delta^1 \in [0,1]^{k'_1\times k_1}$ and $\Delta^2 \in [0,1]^{k'_2\times k_2}$ witnessing respectively $u'\mathcal{R}_1 u$ and $v'\mathcal{R}_2 v$ for the transition vector z'. Define $\Delta = \Delta^1 \odot \Delta^2 \in [0,1]^{(k'_1k'_2)\times(k_1k_2)}$. By Lemma 1, Δ is a correspondence matrix. Moreover, since $\varphi'(u',v')(z')$ holds, there exist $x' \in [0,1]^{k'_1}$ and $y' \in [0,1]^{k'_2}$ such that $\forall i, j,\ z'_{(i,j)} = x'_i y'_j$ and $\varphi'_1(u')(x')$ and $\varphi'_2(v')(y')$.
   - Let $(u'', v'') \in \{1,\dots,k'_1\}\times\{1,\dots,k'_2\}$ such that $z_{(u'',v'')} \neq 0$. By definition of x' and y', this implies that $x'_{u''} \neq 0$ and $y'_{v''} \neq 0$. Thus $\sum_{j=1}^{k_1} \Delta^1_{u''j} = 1$ and $\sum_{j=1}^{k_2} \Delta^2_{v''j} = 1$.

$$\begin{aligned}
\sum_{(r,s)\in\{1,\dots,k_1\}\times\{1,\dots,k_2\}} \Delta_{(u'',v'')(r,s)} &= \sum_{(r,s)\in\{1,\dots,k_1\}\times\{1,\dots,k_2\}} \Delta^1_{u''r}\Delta^2_{v''s}\\
&= \sum_{r=1}^{k_1}\sum_{s=1}^{k_2} \Delta^1_{u''r}\Delta^2_{v''s} = \left(\sum_{r=1}^{k_1}\Delta^1_{u''r}\right)\left(\sum_{s=1}^{k_2}\Delta^2_{v''s}\right) = 1.
\end{aligned}$$

   - Let $z = z'\Delta \in [0,1]^{(k_1k_2)}$. Notice that $z = (x'\Delta^1) \otimes (y'\Delta^2)$.
     Let $x = x'\Delta^1$ and $y = y'\Delta^2$. Since $u'\mathcal{R}_1 u$ and $v'\mathcal{R}_2 v$, we have $\varphi_1(u)(x)$ and $\varphi_2(v)(y)$. Thus $\varphi(u,v)(z'\Delta)$.
   - Let $u'', v'', u'''v'''$ such that $\Delta_{(u'',v'')(u''',v''')} \neq 0$. By definition, this implies that ${\Delta^1}_{u''u'''} \neq 0$ and ${\Delta^2}_{v''v'''} \neq 0$, and as a consequence $(u'', v'')\mathcal{R}(u''', v''')$.

We conclude that $\mathcal{R}$ is a weak refinement relation. Since $(o'_1, o'_2)\mathcal{R}(o_1, o_2)$, we have $S' \preceq S$. The second part of the theorem follows, as satisfaction is a special case of the refinement. □

As alphabets of composed CMCs have to be disjoint, the parallel composition cannot synchronize the components on state valuations like it is typically done for other (non-probabilistic) models. However, synchronization can be introduced by conjoining the parallel composition with a *synchronizer*—a single-state CMC whose valuation function relates the atomic propositions of the composed CMCs.

Example. CMC $S \parallel S'$ of Fig. 12b is synchronized with the synchronizer Sync given in Fig. 12c. Sync removes from $S \parallel S'$ all the valuations that do not satisfy $(a = d) \wedge (b = \neg c)$. The result is given in Fig. 12d. Observe that an inconsistency appears in state (1, 1). Indeed, there is no implementation of the two CMCs that can synchronize in the prescribed way. In general inconsistencies like this one can be uncovered by applying the pruning operator, which would return an empty specification. So synchronizers enable discovery of incompatibilities between component specifications in the same way as it is known for non-probabilistic specification models.

Synchronization is associative with respect to parallel composition, which means that the order of synchronization and parallel composition is inessential for final functionality of the system.

Theorem 18. *Let S_1, S_2 and S_3 be three CMCs with pairwise disjoint sets of propositions A_1, A_2 and A_3. Let Sync_{123} be a synchronizer over $A_1 \cup A_2 \cup A_3$ and let Sync_{12} be the same synchronizer with its set of propositions restricted to $A_1 \cup A_2$. The following holds $[\![[((S_1 \parallel S_2) \wedge \mathsf{Sync}_{12}) \parallel S_3] \wedge \mathsf{Sync}_{123}]\!] = [\![(S_1 \parallel S_2 \parallel S_3) \wedge \mathsf{Sync}_{123}]\!]$.*

Proof. We first prove the following statement. Let S_1 and S_2 be two CMCs with disjoint sets of atomic propositions A_1 and A_2. Let Sync_1 be a synchronizing vector on A_1. We have $(S_1 \parallel S_2) \wedge \mathsf{Sync}_1 = (S_1 \wedge \mathsf{Sync}_1) \parallel S_2$.

First, remember that synchronizers are single-state CMCs, with a single transition taken with probability 1. As a consequence, computing the conjunction with a synchronizer preserves the structure of any CMC. The only change lies in the sets of valuations.

Let p be a state of S_1 and q be a state of S_2. We have $(V_1(p) \cup V_2(q)) \cap V_{\mathsf{Sync}_1}\uparrow^{A_1 \cup A_2} = (V_1(p) \cap V_{\mathsf{Sync}_1}) \cup V_2(q)$. As a consequence, the valuations of $(S_1 \wedge \mathsf{Sync}_1) \parallel S_2$ are the same as the valuations of $(S_1 \parallel S_2) \wedge \mathsf{Sync}_1$.

By monotonicity of conjunction, we have $(S_1 \parallel S_2) \wedge \mathsf{Sync}_{12} \preceq (S_1 \parallel S_2)$. By Theorem 17, it is implied that $[((S_1 \parallel S_2) \wedge \mathsf{Sync}_{12}) \parallel S_3] \wedge \mathsf{Sync}_{123} \preceq [S_1 \parallel S_2 \parallel S_3] \wedge \mathsf{Sync}_{123}$, and finally $[\![[((S_1 \parallel S_2) \wedge \mathsf{Sync}_{12}) \parallel S_3] \wedge \mathsf{Sync}_{123}]\!] \subseteq [\![[S_1 \parallel S_2 \parallel S_3] \wedge \mathsf{S}_{123}]\!]$.

We now prove that $[S_1 \parallel S_2 \parallel S_3] \wedge \mathsf{Sync}_{123} \preceq [((S_1 \parallel S_2) \wedge \mathsf{Sync}_{12}) \parallel S_3] \wedge \mathsf{Sync}_{123}$. By monotonicity of conjunction, we have $[S_1 \parallel S_2 \parallel S_3] \wedge \mathsf{Sync}_{123} \preceq [S_1 \parallel S_2 \parallel S_3] \wedge \mathsf{Sync}_{12} \wedge \mathsf{Sync}_{123}$. Moreover, by the statement proved above, we have $[S_1 \parallel S_2 \parallel S_3] \wedge \mathsf{Sync}_{12} \preceq ((S_1 \parallel S_2) \wedge \mathsf{Sync}_{12}) \parallel S_3$. As a consequence, we have $[S_1 \parallel S_2 \parallel S_3] \wedge \mathsf{Sync}_{123} \preceq [((S_1 \parallel S_2) \wedge \mathsf{Sync}_{12}) \parallel S_3] \wedge \mathsf{Sync}_{123}$, and thus $[\![[S_1 \parallel S_2 \parallel S_3] \wedge \mathsf{Sync}_{123}]\!] \subseteq [\![[((S_1 \parallel S_2) \wedge \mathsf{Sync}_{12}) \parallel S_3] \wedge \mathsf{Sync}_{123}]\!]$. □

Finally, synchronized parallel composition also supports component-based refinement in the style of Theorem 17.

Theorem 19. *If S'_1, S'_2, S_1, S_2 are CMCs, Sync is a synchronizer and $S'_1 \preceq S_1$ and $S'_2 \preceq S_2$, then $(S'_1 \parallel S'_2) \wedge \mathsf{Sync} \preceq (S_1 \parallel S_2) \wedge \mathsf{Sync}$.*

Consequently, a modeler can continue independent refinement of specifications under synchronization, knowing that the original synchronized specification will not be violated. The theorem is a direct corollary of precongruence (Theorem 17) and monotonicity of conjunction (follows from Theorem 14).

6.1. On comparing conjunction and parallel composition

We now compare conjunction and parallel composition with respect to implementation set inclusion. We shall see that if the two operations are defined on CMCs with independent sets of valuations, then parallel composition refines conjunction; the opposite does not hold. We first show that parallel composition refines conjunction.

Theorem 20. *Let S_1 and S_2 be consistent CMCs with $A_1 \cap A_2 = \emptyset$. It holds that $S_1 \parallel S_2 \preceq S_1 \wedge S_2$.*

Proof. Let $S_1 = \langle\{1, \dots, k_1\}, o_1, \varphi_1, A_1, V_1\rangle$ and $S_2 = \langle\{1, \dots, k_2\}, o_2, \varphi_2, A_2, V_2\rangle$ be CMCs. Consider their parallel composition $S_1 \parallel S_2 = \langle\{1, \dots, k_1\} \times \{1, \dots, k_2\}, (o_1, o_2), \varphi^{\parallel}, A, V^{\parallel}\rangle$ and their conjunction $S_1 \wedge S_2 = \langle\{1, \dots, k_1\} \times \{1, \dots, k_2\}, (o_1, o_2), \varphi^{\wedge}, A, V^{\wedge}\rangle$, where $A = A_1 \cup A_2$. We build a refinement relation $\mathcal{R}$ on $(\{1, \dots, k_1\} \times \{1, \dots, k_2\}) \times (\{1, \dots, k_1\} \times \{1, \dots, k_2\})$ as $(u, v)\mathcal{R}(u', v')$ if and only if $u = u'$ and $v = v'$.

Let $(u, v) \in \{1, \dots, k_1\} \times \{1, \dots, k_2\}$ such that $(u, v)\mathcal{R}(u, v)$. We now show that $\mathcal{R}$ is a refinement relation:

1. By construction, we have that $V^{\parallel}((u, v)) = \{Q_1 \cup Q_2 \mid Q_1 \in V_1(u), Q_2 \in V_2(v)\}$. Moreover, since $A_1 \cap A_2 = \emptyset$, we have that $V^{\wedge}((u, v)) = V_1(u)\uparrow^A \cap V_2(v)\uparrow^A = \{Q_1 \cup Q_2 \mid Q_1 \in V_1(u), Q_2 \in V_2(v)\}$. Thus $V^{\parallel}((u, v)) = V^{\wedge}((u, v))$.
2. Let $z = (z_{1,1}, z_{1,2}, \dots, z_{k_1,k_2}) \in [0, 1]^{k_1 k_2}$ such that $\varphi^{\parallel}((u, v))(z)$ holds. Define the correspondence matrix $\Delta \in [0, 1]^{(k_1 k_2) \times (k_1 k_2)}$ as the matrix with $\Delta_{(u,v),(u,v)} = 1$ if $z_{u,v} \neq 0$ and 0 otherwise. Observe that $z\Delta = z$.
   - Trivially, by construction, for all $(i, j) \in \{1, \dots, k_1\} \times \{1, \dots, k_2\}$ such that $z_{i,j} \neq 0$, we have that $\sum_{i',j'} \Delta_{(i,j),(i',j')} = 1$.
   - We prove that $\varphi^{\wedge}((u, v))(z)$ holds: By hypothesis, $\varphi^{\parallel}((u, v))(z)$ holds. So there exist $x \in [0, 1]^{k_1}$ and $y \in [0, 1]^{k_2}$ such that $\varphi_1(u)(x)$ holds, $\varphi_2(v)(y)$ holds and for all $i \in \{1, \dots, k_1\}$ and $j \in \{1, \dots, k_2\}$, we have $z_{i,j} = x_i y_j$. As a consequence, we have $\sum_{i=1}^{k_1} z_{i,j} = y_j$ for all $j \in \{1, \dots, k_2\}$ and $\sum_{j=1}^{k_2} z_{i,j} = x_i$ for all $i \in \{1, \dots, k_1\}$. Since both $\varphi_1(u)(x)$ and $\varphi_2(v)(y)$ hold, we have that $\varphi^{\wedge}((u, v))(z\Delta)$ holds.
   - By construction of Δ, $\Delta_{(u,v),(u',v')} \neq 0$ implies that $u = u'$ and $v = v'$, and therefore implies $(u, v)\mathcal{R}(u', v')$.

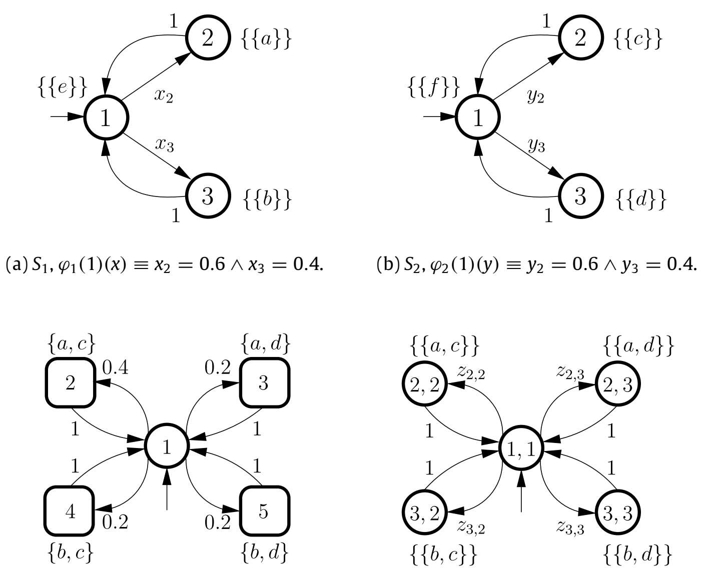
We conclude that $\mathcal{R}$ is a refinement relation since $(o_1, o_2)\mathcal{R}(o_1, o_2)$. We have shown that $S_1 \parallel S_2 \preceq S_1 \wedge S_2$. □

A direct consequence of the above theorem is that any model of the parallel composition is a model for the conjunction, i.e.,$[\![S_1 \parallel S_2]\!] \subseteq [\![S_1 \wedge S_2]\!]$. We now show that the opposite inclusion does not hold.

Theorem 21. *Let S_1 and S_2 be consistent CMCs with $A_1 \cap A_2 = \emptyset$. It holds that $[\![S_1 \wedge S_2]\!] \not\subseteq [\![S_1 \parallel S_2]\!]$.*

Proof. We establish the proof by providing, in Fig. 13, two CMCs S_1 and S_2 and a MC I, such that $I \models S_1 \wedge S_2$ and $I \not\models S_1 \parallel S_2$.

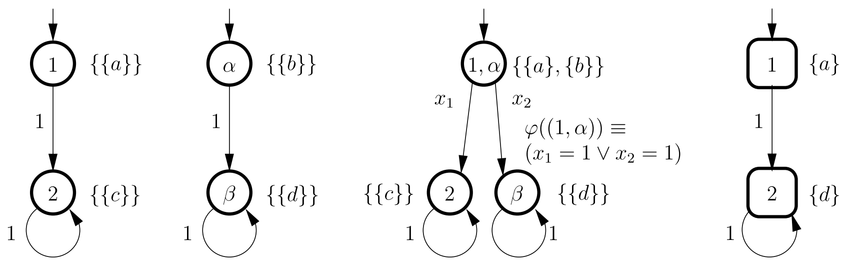
The common structure of conjunction and parallel composition is shown in Fig. 13d. The constraint functions differ. According to the definitions of conjunction and parallel composition, we have $\varphi^{\wedge}(1, 1)(z) \equiv z_{2,2} + z_{2,3} = z_{2,2} + z_{3,2} =$



(a) S_1, $\varphi_1(1)(x) \equiv x_2 = 0.6 \wedge x_3 = 0.4$. (b) S_2, $\varphi_2(1)(y) \equiv y_2 = 0.6 \wedge y_3 = 0.4$.

(c) I. (d) Common structure of $S_1 \wedge S_2$ and $S_1 \parallel S_2$.

Fig. 13. Conjunction versus parallel composition: $I \models S_1 \wedge S_2$ but $I \not\models S_1 \parallel S_2$.



(a) CMC S_1. (b) CMC S_2. (c) CMC S_3 such that $[\![S_1]\!] \cup [\![S_2]\!] \subsetneq [\![S_3]\!]$. (d) MC I.

Fig. 14. CMCs not closed under disjunction: $I \models S_3$, but $I \not\models S_1$ and $I \not\models S_2$.

$0.6 \wedge z_{3,2} + z_{3,3} = z_{2,3} + z_{3,3} = 0.4$ and $\varphi^{\parallel}(1, 1)(z) \equiv z_{2,2} = 0.36 \wedge z_{2,3} = z_{3,2} = 0.24 \wedge z_{3,3} = 0.16$. I satisfies the conjunction, but not the parallel composition, since the probability mass 0.4 of going to state 2 in I cannot be distributed to (2, 2) of $S_1 \parallel S_2$. □

7. Disjunction and universality

In this section we show that CMCs are not closed under disjunction. We then solve the *universality problem*, that is the problem of deciding whether a CMC admits arbitrary implementations.

7.1. On the existence of a disjunction of CMCs

In this section we discuss the problem of computing a CMC S whose models are the union of the models accepted by two other CMCs $S_1 = \langle\{1, \ldots, k_1\}, o_1, \varphi_1, A_1, V_1\rangle$ and $S_2 = \langle\{1, \ldots, k_2\}, o_2, \varphi_2, A_2, V_2\rangle$. In general, such a CMC may not exist. Indeed, assume that S_1 and S_2 have disjoint initial state valuations, and that the constraint functions of o_1 and o_2 do not share the same set of satisfying probability vectors. The initial state o of any specification representing the union could take valuations admissible according to o_1 and a distribution M_o according to o_2 (but not o_1). That is, we cannot express the link between a choice of the valuation of the initial state and a probability distribution.

This is illustrated in Fig. 14. S_1 admits implementations with valuation a in the initial state, and c afterward. S_2 admits implementations with b in the initial state, and d afterward. Any CMC representing the disjunction would have to admit a or b in the initial state, like the potential candidate S_3 in Fig. 14c. Unfortunately, the implementation MC I in Fig. 14d satisfies S_3 but it is a model of neither S_1 and S_2, as it combines the initial state valuation of the former, with behavior of the latter.

However, if S_1 and S_2 have the same initial state valuation, then we can explicitly construct a CMC whose set of implementations is the union of the sets of implementations of S_1 and S_2. This CMC is called the *disjunction* of S_1 and S_2, and denoted $S_1 \vee S_2$.

Let $S_1 = \langle\{1,\ldots,k_1\}, o_1, \varphi_1, A_1, V_1\rangle$ and $S_2 = \langle\{1,\ldots,k_2\}, o_2, \varphi_2, A_2, V_2\rangle$ be CMCs such that $V_1(o_1) = V_2(o_2)$. Then define $S_1 \vee S_2 = \langle Q, 0, \varphi, A, V\rangle$, where $Q = \{0, 1, \ldots, k_1, k_1+1, \ldots, k_1+k_2\}$, $A = A_1 \cup A_2$ and

$$V(i) = \begin{cases} V_1(o) & \text{if } i = 0, o = o_1 = o_2 \\ V_1(i) & \text{if } i \in \{1,\ldots,k_1\} \\ V_2(i-k_1) & \text{if } i \in \{k_1+1,\ldots,k_1+k_2\}. \end{cases}$$

The constraint function $\varphi : Q \to [0,1]^{k_1+k_2+1} \to \{0,1\}$ is given by:

$$\varphi(i)(x_0, x_1, \ldots, x_{k_1}, x_{k_1+1}, \ldots, x_{k_1+k_2}) \equiv \begin{cases} \left(\varphi_1(i)(x_1,\ldots,x_{k_1}) \vee \varphi_2(i)(x_{k_1+1},\ldots,x_{k_1+k_2})\right) \wedge \sum_{i=0}^{k_1+k_2} x_i = 1 & \text{if } i = 0 \\ \varphi_1(i)(x_1,\ldots,x_{k_1}) \wedge \left(x_0 + \sum_{i=k_1+1}^{k_1+k_2} x_i = 0\right) & \text{if } 1 \le i \le k_1 \\ \varphi_2(i)(x_{k_1+1},\ldots,x_{k_1+k_2}) \wedge \sum_{i=0}^{k_1} x_i = 0 & \text{if } k_1+1 \le i \le k_1+k_2. \end{cases}$$

7.2. The universality problem for CMCs

Consider the problem of deciding whether a CMC S admits all models defined over a set of atomic propositions A. This problem can be reduced to checking whether the set of implementations of the universal CMCs Univ^A representing all models over A is included in the set of implementations of S. The CMC Univ^A is defined as $\mathsf{Univ}^A = \langle\{1\}, 1, \varphi, A, V\rangle$, where $\varphi(1)(x) \equiv 1$ and $V(1) = 2^A$.

Theorem 22. *Let* $\mathsf{Univ}^A = \langle\{1\}, 1, \varphi, A, V\rangle$ *be the universal CMC on the set of atomic propositions A and let* $I = \langle\{1,\ldots,n\}, o, M, A_I, V_I\rangle$ *be any implementation such that* $A \subseteq A_I$. *We have that* $I \models \mathsf{Univ}^A$.

Proof. Construct the relation $\mathcal{R} = \{1,\ldots,n\} \times \{1\}$. We show that $\mathcal{R}$ is a satisfaction relation: Let $i \in \{1,\ldots,n\}$ such that $i\mathcal{R}1$.

1. It is clear that $V_I(i)\downarrow_A \in V(1) = 2^A$.
2. Consider M_i. We build a correspondence matrix $\Delta \in [0,1]^{n\times 1}$ such that $\Delta_{j1} = 1$ if $M_{ij} > 0$, and 0 otherwise.
   - By construction, $\Delta_{j1} = 1$ for all j such that $M_{ij} > 0$.
   - Since $M_i\Delta = 1$, $\varphi(1)(M_i\Delta)$ holds.
   - Let i' such that $\Delta_{i',1} > 0$. By construction of $\mathcal{R}$, $i'\mathcal{R}1$.

We conclude that $\mathcal{R}$ is a satisfaction relation, since $o\mathcal{R}1$, and thus, $I \models \mathsf{Univ}^A$. □

We now switch to the problem of deciding whether the union of two CMCs S_1 and S_2 is universal. Despite the fact that CMCs are not closed under union, this problem has a relatively simple solution. The idea is to create a new initial state with a fresh atomic proposition $\lambda \notin A$ and then redistribute the entire probability mass to the original initial state. Formally:

Definition 23. For a CMC $S = \langle\{1,\ldots,k\}, o, \varphi, A, V\rangle$ and an atomic proposition $\lambda \notin A$ define the state-extended CMC $S^x = \langle\{1,\ldots,k,o'\}, o', \varphi', A', V'\rangle$, where $A' = A \cup \{\lambda\}$, $V'(o') = \{\{\lambda\}\}$ and $V'(i) = V(i)$ for all $i \in \{1,\ldots,k\}$. The constraint function is given by:

$$\varphi'(i)(x) \equiv \begin{cases} x_o = 1 \wedge \left(\sum_{i=1}^{k} x_i = 0\right) & \text{if } i = o' \\ \varphi(i)(x_1,\ldots,x_k) \wedge x_{o'} = 0 & \text{for } i \in \{1,\ldots,k\}. \end{cases}$$

Fig. 15 gives an example. The union of the state-extended versions of S_1 and S_2 can now be computed and compared to the state-extended version of Univ^A. It is obvious that all the implementations of the state-extended version of a given CMC C are state-extended versions of implementations of C.

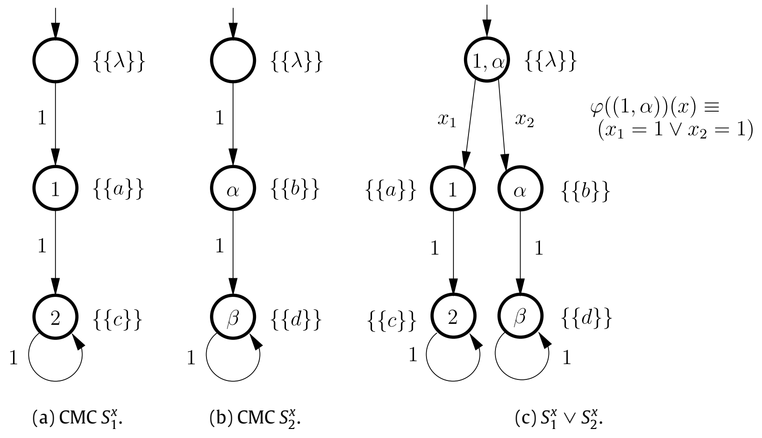


Fig. 15. State-extended CMCs of Fig. 14 and their disjunction.

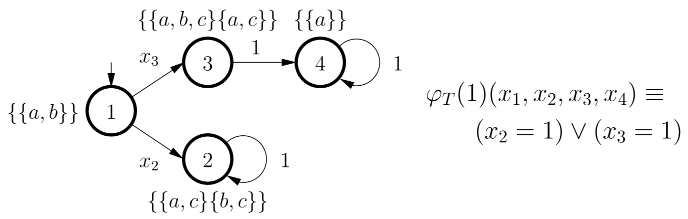


Fig. 16. A deterministic CMC cannot express the implementation set of T.

8. Deterministic CMCs

Clearly, if all implementations of a specification S_1 also implement a specification S_2, then the former is a strengthening of the latter. Indeed, S_1 specifies implementations that break no assumptions that can be made about implementations of S_2. Thus implementation set inclusion *is* a desirable refinement for specifications. Unfortunately the decision procedure for implementation set inclusion is more complex and less efficient than the weak and (in particular) the strong refinement. We have seen that the latter two both soundly approximate implementation set inclusion. Had that approximation been complete, we could have a more efficient and simpler to program procedure for implementing the implementation set inclusion.

In this section, we argue that this indeed is the case for an important subclass of specifications: *deterministic CMCs*. We show that for this class strong refinement coincides with the implementation set inclusion. Thus for deterministic CMCs more efficient algorithms exist for establishing the latter. For this reason we will also consider a determinization algorithm for CMCs.

A CMC S is *deterministic* iff for every state i, states reachable from i have pairwise disjoint admissible valuations:

Definition 24. Let $S = \langle\{1, \dots, k\}, o, \varphi, A, V\rangle$ be a CMC. S is *deterministic* iff for all states $i, u, v \in \{1, \dots, k\}$, if there exists $x \in [0, 1]^k$ such that $(\varphi(i)(x) \wedge (x_u \neq 0))$ and $y \in [0, 1]^k$ such that $(\varphi(i)(y) \wedge (y_v \neq 0))$, then we have that $V(u) \cap V(v) = \emptyset$ or $u = v$.

By inspecting the constructions for conjunction and parallel composition, one can see that both trivially preserve determinism.

In Fig. 3a and b, both S_1 and S_2 are deterministic specifications. In particular states 2 and 3, both of which can be reached from state 1 in both CMCs, have disjoint valuation sets. On the other hand, the CMC T given in Fig. 16 is non-deterministic. Indeed, for states 2 and 3, which can both be reached from state 1, we have that $V_T(2) \cap V_T(3) = \{\{a, c\}\} \neq \emptyset$.

Deterministic CMCs are less expressive than non-deterministic ones, in the sense that the same implementation sets sometimes cannot be expressed. Consider again the CMC T given in Fig. 16. The set of implementations of T cannot be represented by a deterministic CMC. Any merging of States 2 and 3 in T would result in a CMC that accepts models where one can loop on valuation $\{a, c\}$ and then accept valuation $\{a\}$ with probability 1. Such a model cannot be accepted by T.

We now present a determinization algorithm that can be applied to any CMC S in single valuation normal form (obtainable by applying the normalization algorithm of Definition 7). This algorithm is based on a subset construction that resembles the classical determinization for automata.

Definition 25. Let $S = \langle \{1, \ldots, k\}, o, \varphi, A, V\rangle$ be a consistent CMC in single valuation normal form. If $Q \subseteq \{1, \ldots, k\}$ and $a \in 2^A$, then define $\mathsf{reach}(Q, a)$ to be the maximal[1] set of states with valuation a that can be reached with a non-zero probability, using a distribution that satisfies the constraint of at least one state of Q. Formally $\mathsf{reach} : 2^{\{1,\ldots,k\}} \times 2^A \mapsto 2^{\{1,\ldots,k\}}$ and

$$\mathsf{reach}(Q, a) = \bigcup \{v \in \{1, \ldots, k\} \mid V(v) = \{a\} \text{ and } \exists u \in Q.\, \exists x \in [0, 1]^k.\, \varphi(u)(x) = 1 \wedge x_v > 0\}.$$

We lift the notion of reachability to all possible valuations as follows:

$$\mathsf{Reach}(Q) = \{\mathsf{reach}(Q, a) \mid a \subseteq A\}.$$

Now define the n-step reachability with

$$\mathsf{Reach}^n(Q) = \mathsf{Reach}^{n-1}(Q) \cup \bigcup_{Q' \in \mathsf{Reach}^{n-1}(Q)} \mathsf{Reach}(Q')$$

and its transitive closure as the fixpoint

$$\mathsf{Reach}^*(Q) = \{Q\} \cup \lim_{n\to\infty} \mathsf{Reach}^n(Q).$$

Observe that for all Q and for all $Q' \in \mathsf{Reach}^*(Q)$, there exists a valuation $a \in 2^A$ such that $V(q) = \{a\}$ for all $q \in Q'$ (by construction).

A deterministic CMC for S is the CMC $\varrho(S) = \langle \{Q_1, \ldots, Q_m\}, Q_{o'}, \varphi', A, V'\rangle$, where $\{Q_1, \ldots, Q_m\} = \mathsf{Reach}^*(\{o\})$, $Q_{o'} = \{o\} \in \{Q_1, \ldots, Q_m\}$ by definition, and φ' and V' are defined as follows:

- for all $Q_i \in \{Q_1, \ldots, Q_m\}$, let $V'(Q_i) = \{a\}$ iff for all $q \in Q_i$, $V(q_i) = \{a\}$ – there always exists exactly one such a by construction, and
- for all $Q_i \in \{Q_1, \ldots, Q_m\}$ and for all $y \in [0, 1]^m$,

$$\varphi'(Q_i)(y_1, \ldots, y_m) = [\forall 1 \le j \le m.\, Q_j \notin \mathsf{Reach}(Q_i) \implies y_j = 0] \wedge$$
$$\bigvee_{q\in Q_i} \exists x \in [0, 1]^k \left[\varphi(q)(x) \wedge \left(\forall 1 \le j \le m.\ Q_j \in \mathsf{Reach}(Q_i) \implies \sum_{q'\in Q_j} x_{q'} = y_j \right) \right].$$

Theorem 26. *Let S be a CMC in* single valuation normal form. *It holds that* $S \preceq_S \varrho(S)$.

Proof. Let $S = \langle \{1, \ldots, k\}, o, \varphi, A, V\rangle$ be a CMC in single valuation normal form. Let $\varrho(S) = \langle \{Q_1, \ldots, Q_m\}, Q_{o'}, \varphi', A, V'\rangle$ be a determinization of S.

Define $\mathcal{R} \subseteq \{1, \ldots, k\} \times \{Q_1, \ldots, Q_m\}$ such that $u\mathcal{R}Q_i \iff u \in Q_i$. We will show that $\mathcal{R}$ is a strong refinement relation. Let u, i such that $u\mathcal{R}Q_i$.

1. By definition, since $u \in Q_i$, we have $V'(Q_i) = V(u)$.
2. Let $\Delta \in [0, 1]^{k\times m}$ such that $\Delta_{vj} = 1$ if $Q_j \in \mathsf{Reach}(Q_i)$ and $v \in Q_j$, and 0 otherwise. We prove that Δ is a correspondence matrix.

   Suppose that there exist $1 \le v \le k$ and $1 \le j \ne l \le m$ such that $\Delta_{vj} = \Delta_{vl} = 1$. Then we know that $v \in Q_j$ and $v \in Q_l$, and that both Q_j and Q_l are in $\mathsf{Reach}(Q_i)$. This violates the definition of $\mathsf{Reach}(Q_i)$ (each of its sets must be maximal). As a consequence, for all $1 \le v \le k$, we have $\sum_{j=1}^m \Delta_{vj} \le 1$ and Δ is a correspondence matrix.
   - Let $x \in [0, 1]^k$ such that $\varphi(u)(x)$. Let $1 \le v \le k$ such that $x_v > 0$. Since $x_v > 0$ and $u \in Q_i$, we know that $\mathsf{Reach}(Q_i) \ne \emptyset$. Moreover, there exists $1 \le j \le m$ such that $Q_j \in \mathsf{Reach}(Q_i)$ and $v \in Q_j$. As a consequence, $\Delta_{vj} = 1$, and $\sum_{l=1}^m \Delta_{vl} = 1$.
   - Let $y = x\Delta$. We prove that $\varphi'(y)$ holds.
     - Since $\Delta_{vj} = 0$ for all $Q_j \notin \mathsf{Reach}(Q_i)$, we have that $y_j = 0$ for all $Q_j \notin \mathsf{Reach}(Q_i)$.
     - There exist $q \in Q_i$, namely u, and $x \in [0, 1]^k$ defined above, such that $\varphi(u)(x)$ holds, and by definition, if $Q_j \in \mathsf{Reach}(Q_i)$, then for all $q' \in Q_j$, we have $\Delta_{q'j} = 1$. As a consequence, $y_j = \sum_{r=1}^k x_r\Delta_{rj} = \sum_{q'\in Q_j} x_{q'}$.

     Thus $\varphi'(x\Delta)$ holds.
   - If $\Delta_{vj} > 0$, then we have that $v \in Q_j$ by definition, thus $v\mathcal{R}Q_j$.

Finally, $\mathcal{R}$ is a strong refinement relation, and $o \in Q_{o'} = \{o\}$, thus S strongly refines $\varrho(S)$. □

This character of determinization resembles the known determinization algorithms for modal transition systems [28].

In Theorem 11 we have shown that weak (and thus also strong) refinement is sound with respect to implementation set inclusion. We now state one of the main theorems of the paper. Weak refinement is complete with respect to implementation set inclusion for deterministic CMCs.

[1] Maximality is understood with respect to set inclusion here. It captures the fact that we want all such states.

Theorem 27. *Let $S_1 = \langle\{1,\ldots,k_1\}, o_1, \varphi_1, A_1, V_1\rangle$ and $S_2 = \langle\{1,\ldots,k_2\}, o_2, \varphi_2, A_2, V_2\rangle$ be two locally consistent deterministic CMCs with single admissible valuation in initial state and $A_2 \subseteq A_1$. We have $[\![S_1]\!] \subseteq [\![S_2]\!] \Rightarrow S_1 \preceq S_2$.*

Proof. First, since any consistent CMC with a single valuation in the initial state can be normalized (see Theorem 8), without change of the implementation set, we assume that S_1 and S_2 are actually in single valuation normal form.

Let $S_1 = \langle\{1,\ldots,k_1\}, o_1, \varphi_1, A_1, V_1\rangle$ and $S_2 = \langle\{1,\ldots,k_2\}, o_2, \varphi_2, A_2, V_2\rangle$ be two consistent and deterministic CMCs in single valuation normal form such that $A_2 \subseteq A_1$ and $[\![S_1]\!] \subseteq [\![S_2]\!]$.

First, notice that $S_1 \preceq S_2 \iff S_1' = \langle\{1,\ldots,k_1\}, o_1, \varphi_1, A_2, V_1{\downarrow_{A_2}}\rangle \preceq S_2$. It is thus safe to suppose that $A_1 = A_2 = A$. Similarly, if $I = \langle\ldots, A_I, V_I\rangle$ is a MC, we have $I \models S_1 \iff I' = \langle\ldots, A_1, V_I{\downarrow_A}\rangle \models S_1$. As a consequence, it is also safe to suppose that implementations have the same set of atomic propositions as S_1 and S_2.

In the following we will also rely on the local consistency of the two CMCs, which implies that for every state of a CMC there exists a MC satisfying it. Thanks to Theorem 5, local consistency is not a real limitation, as the specifications can always be pruned without loss of implementations.

In the proof we use the following notation: given a CMC S and its state o we write (S, o) to denote a new CMC created from S by assuming o as its initial state.

The proof is structured as a usual coinductive argument, starting with the presentation of a candidate relation $\mathcal{R}$ and then continuing with evidence that $\mathcal{R}$ is indeed a refinement relation witnessing the refinement of S_2 by S_1. The argument is essentially standard until the last step, marked with a ⋆ below, where we need to rely on the determinism of S_2 and an argument by contradiction to conclude.

Let $\mathcal{R} \subseteq \{1,\ldots,k_1\} \times \{1,\ldots,k_2\}$ be the following binary relation on states:

$$\mathcal{R} = \{(v, u) \mid \text{For all } I.\, I \models (S_1, v) \text{ implies } I \models (S_2, u)\}\,. \tag{4}$$

Consider v and u such that $v\mathcal{R}u$ and check that conditions of Definition 4 hold.

1. By local consistency of S_1 there exists a MC $I = \langle\{1,\ldots,n\}, p, M, A, V\rangle$ such that $I \models (S_1, v)$, and, since $v\mathcal{R}u$, then also $I \models (S_2, u)$. Thus $V(p) \in V_1(v)$ and $V(p) \in V_2(u)$. As S_1 and S_2 are in single valuation normal form, $V_1(v)$ and $V_2(u)$ are singletons, so $V_1(v) = V_2(u)$.
2. Consider a probability distribution vector $x \in [0,1]^{k_1}$ such that $\varphi_1(v)(x)$. We want to build a correspondence matrix $\Delta \in [0,1]^{k_1 \times k_2}$ such that $\varphi_2(u)(x\Delta)$ holds. To this end, we build another implementation MC $I = \langle\{1,\ldots,k_1\}, v, M, A, V\rangle$ such that for all $1 \le w \le k_1$,
   (i) $V(w)$ is the only valuation such that $V_1(w) = \{V(w)\}$; The existence of $V(w)$ is warranted by the normal form and local consistency of S_1.
   (ii) If $w \ne v$, the row M_w is any solution of $\varphi_1(w)$. One exists for each state w of S_1 because S_1 is locally consistent.
   (iii) $M_v = x$.

   When necessary, we will address state w of I as w_I to differentiate it from state w of S_1. The MC I clearly satisfies (S_1, v) as witnessed by the identity satisfaction relation $\mathcal{R}_1$. By hypothesis, we thus have $I \models (S_2, u)$, as $v\mathcal{R}u$. Let $\mathcal{R}_2$ be the relation witnessing $I \models (S_2, u)$, and let Δ_2 be the correspondence matrix witnessing $v_I\mathcal{R}_2u$. We will now show the correspondence matrix Δ witnessing the weak refinement invariant for $v\mathcal{R}u$. Let $\Delta = \Delta_2$.
   - $\forall 1 \le i \le k_1, x_i \ne 0 \Rightarrow \sum_{j=1}^{k_2} \Delta_{ij} = 1$, because $\Delta = \Delta_2$, which is a correspondence matrix witnessing satisfaction for the same probability vector $M_v = x$.
   - $\varphi_2(u)(x\Delta)$ holds for the same reason.
   - It remains to show that if $x_{v'} \ne 0$ and $\Delta_{v'u'} \ne 0$, then $v'\mathcal{R}u'$. This argument occupies the remainder of the proof.

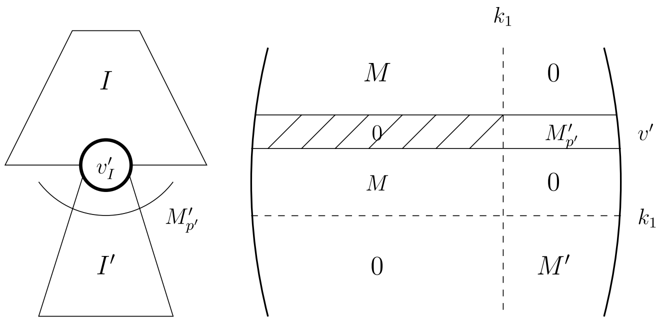
⋆ Assume v', u' as above. By definition of I and Δ we have that $(I, v_I') \models (S_1, v')$ and $(I, v_I') \models (S_2, u')$. We want to prove not only for (I, v_I') but also for all implementations I' such that $I' \models (S_1, v')$ that $I' \models (S_2, u')$. This argument proceeds ad absurdum.

Suppose this is not the case: there exists a MC $I' = \langle\{1,\ldots,n\}, p', M', A, V'\rangle$ such that $I' \models (S_1, v')$ and $I' \not\models (S_2, u')$. Let $\mathcal{R}'$ be the satisfaction relation witnessing $I' \models (S_1, v')$. We will use this implementation to construct an implementation $\widehat{I}$ of (S_1, v) which also satisfies (S_2, u). Since $\widehat{I}$ will embed I', and S_2 is deterministic, we will be able to obtain that $I' \models (S_2, u')$, which is a contradiction. Thus I' cannot exist, and indeed $v'\mathcal{R}u'$.

Below we construct $\widehat{I}$ and argue that $\widehat{I} \models (S_1, v)$ and thus $\widehat{I} \models (S_2, u)$. In the last part of the proof, marked with a ⊚, we argue that $I' \models (S_2, u')$, which leads to a contradiction, concluding the proof.

Let $\widehat{I} = \langle\{1,\ldots,k_1, k_1+1,\ldots,k_1+n\}, v, \widehat{M}, A, \widehat{V}\rangle$, where, among others, v and n are defined above. Intuitively, the first k_1 states correspond to I and the next n states to I'. The state v_I' will be the link between the two and its outgoing transitions will be the ones of p', the state of I'. The construction is illustrated in Fig. 17. The left part of the figure shows the general structure of $\widehat{I}$, the right part shows the composition of its transition matrix. Formally, we define the transition matrix $\widehat{M}$ as follows:

$$\widehat{M}_{ij} = \begin{cases} M_{ij} & \text{if } 1 \le i,j \le k_1 \text{ and } i \ne v' \\ 0 & \text{if } 1 \le j \le k_1 \text{ and } i = v' \\ 0 & \text{if } 1 \le i \le k_1 \text{ and } i \ne v',\ j > k_1 \\ M'_{p'(j-k_1)} & \text{if } j > k_1 \text{ and } i = v' \\ 0 & \text{if } i > k_1 \text{ and } 1 \le j \le k_1 \\ M'_{(i-k_1)(j-k_1)} & \text{if } i,j > k_1. \end{cases}$$



(a) The MC $\widehat{I}$. (b) The transition matrix $\widehat{M}$.

Fig. 17. Visualization of the construction of $\widehat{I}$ for the proof of Theorem 27.

Furthermore:

$$\widehat{V}(i) = \begin{cases} V(i) & \text{for } i \le k_1 \\ V'(i-k_1) & \text{for } i > k_1. \end{cases}$$

We want to show that $\widehat{I} \models (S_1, v)$ first. Consider the following relation $\widehat{\mathcal{R}} \subseteq \{1,\ldots,k_1+n\} \times \{1,\ldots,k_1\}$, between the states of $\widehat{I}$ and the states of S_1:

$$\widehat{\mathcal{R}} = \{(q,w) \in \mathcal{R}_1 \mid q \neq v'\} \cup \{(q,w) \mid (q-k_1)\mathcal{R}'w\} \cup \{(v',w) \mid p'\mathcal{R}'w\}. \tag{5}$$

Intuitively, $\widehat{\mathcal{R}}$ is equal to $\mathcal{R}_1$ for the states $q \le k_1$, except v', and equal to $\mathcal{R}'$ for the states $q > k_1$. The states related to $v'_{\widehat{I}}$ are the ones that were related to p' with $\mathcal{R}'$. We will show that $\widehat{\mathcal{R}}$ is a satisfaction relation between $\widehat{I}$ and (S_1, v).

Let q, w be states of $\widehat{I}$ and S_1 respectively, such that $q\widehat{\mathcal{R}}w$. For all the pairs where $q \neq v'_{\widehat{I}}$, the conditions of the satisfaction relation still hold because they held for $\mathcal{R}_1$ if $q \le k_1$ and for $\mathcal{R}'$ otherwise ($\mathcal{R}_1 \subseteq \widehat{\mathcal{R}}$ by construction, since $v'_{\widehat{I}}\widehat{\mathcal{R}}v'$ and moreover v' is the only state to which $v'_{\widehat{I}}$ was related in the identity relation $\mathcal{R}_1$). It remains to check the conditions for the pairs where $q = v'_{\widehat{I}}$, as this is the only state with a new behavior with respect to I and I'. So consider a state w of S_1 such that $v'_{\widehat{I}}\widehat{\mathcal{R}}w$.

1. Because v'_I and $p'_{I'}$ are both related to v' (respectively in $\mathcal{R}_1$ and in $\mathcal{R}'$) it is clear that $\widehat{V}(v'_{\widehat{I}}) = \widehat{V}(p')$. As $p'\mathcal{R}'w$, we know that $V'(p') \in V_1(w)$. Thus, $\widehat{V}(v'_{\widehat{I}}) \in V_1(w)$.
2. Let the correspondence matrix Δ' witness $p'\mathcal{R}'w$. Let $\widehat{\Delta} \in [0,1]^{(k_1+n)\times k_1}$ such that $\widehat{\Delta}_{ij} = 0$ if $i \le k_1$, and $\widehat{\Delta}_{ij} = \Delta'_{(i-k_1)j}$ otherwise.
   - We want to show that if $\widehat{M}_{(v'_{\widehat{I}})(w')} \neq 0$, then $\sum_{j=1}^{k_1} \widehat{\Delta}_{w'j} = 1$. We know that $\widehat{M}_{(v'_{\widehat{I}})(w')} = 0$ if $w' \le k_1$. Take $w' > k_1$ such that $\widehat{M}_{(v'_{\widehat{I}})(w')} \neq 0$. Then we know that $\widehat{M}_{(v'_{\widehat{I}})(w')} = M'_{p'(w'-k_1)}$. Because $\mathcal{R}'$ is a satisfaction relation, it implies that $\sum_{j=1}^{k_1} \Delta'_{(w'-k_1)j} = 1$. Thus, $\sum_{j=1}^{k_1} \widehat{\Delta}_{w'j} = \sum_{j=1}^{k_1} \Delta'_{(w'-k_1)j} = 1$.
   - We want to show now that $\varphi_1(w)(\widehat{M}_{v'_{\widehat{I}}}\widehat{\Delta})$ holds. Let $1 \le j \le k_1$. We have:

$$\left[\widehat{M}_{v'_{\widehat{I}}}\widehat{\Delta}\right]_j = \sum_{l=1}^{k_1+n} \widehat{M}_{(v'_{\widehat{I}})l}\widehat{\Delta}_{lj} = 0 + \sum_{l=k_1+1}^{k_1+n} \widehat{M}_{(v'_{\widehat{I}})l}\widehat{\Delta}_{lj} = \sum_{l=1}^{n} M'_{p'l}\Delta'_{lj} = \left[M'_{p'}\Delta'\right]_j .$$

   As a consequence, $\widehat{M}_{v'_{\widehat{I}}}\widehat{\Delta} = M'_{p'}\Delta'$. Since Δ' is a witness of $p'\mathcal{R}'w$, $\varphi_1(w)(M'_{p'}\Delta')$ holds. So does $\varphi_1(w)(\widehat{M}_{v'_{\widehat{I}}}\widehat{\Delta})$.
   - We want to show that if $\widehat{M}_{(v'_{\widehat{I}})q} \neq 0$ and $\widehat{\Delta}_{qw'} \neq 0$, then $q\widehat{\mathcal{R}}w'$. We only need to consider $q > k_1$ (since otherwise $\widehat{M}_{(v'_{\widehat{I}})q} = 0$) and w' such that $\widehat{\Delta}_{qw'} \neq 0$. In this case, $\widehat{M}_{(v'_{\widehat{I}})q} = M'_{p'(q-k_1)} \neq 0$ and $\Delta'_{(q-k_1)w'} \neq 0$. As Δ' is a witness of $p'\mathcal{R}'w$, it has to be that $(q-k_1)\mathcal{R}'w'$, which implies, by definition of $\widehat{\mathcal{R}}$, that $q\widehat{\mathcal{R}}w'$.

So we conclude that $\widehat{I} \models (S_1, v)$, and thus also $\widehat{I} \models (S_2, u)$ since $v\mathcal{R}u$.

Finally, to reach the contradiction, we show that the above implies $I' \models (S_2, u')$. Since $\widehat{I} \models (S_2, u)$ there exists $\Delta'' \in [0,1]^{(k_1+n)\times k_2}$ such that $\varphi_2(u)(\widehat{M}_{v_{\widehat{I}}}\Delta'')$ holds.

(A) Consider $u'' \neq u'$ such that $V_2(u'') = V_2(u')$. Due to the determinism of S_2, and to the fact that u' is accessible from u, we have $[\widehat{M}_{v_{\widehat{I}}}\Delta'']_{u''} = 0$. Otherwise $\varphi_2(u)$ would be violated. Since $\widehat{M}_{(v_{\widehat{I}})(v'_{\widehat{I}})} \neq 0$ and $\widehat{M}_{(v_{\widehat{I}})(v'_{\widehat{I}})}\Delta''_{(v'_{\widehat{I}})u''}$ is part of $[\widehat{M}_{v_{\widehat{I}}}\Delta'']_{u''}$, we must have $\Delta''_{(v'_{\widehat{I}})u''} = 0$.

(B) Consider u''' such that $V(u''') \neq V(u')$. It is clear that $\Delta''_{(v'_{\widehat{I}})u'''} = 0$ since Δ'' is witnessing satisfaction between $\widehat{I}$ and S_2.

(C) Moreover, we know that $\widehat{M}_{(v_{\widehat{I}})(v'_{\widehat{I}})} \neq 0$. Thus, $\sum_{j=1}^{k_2} \Delta''_{v'_{\widehat{I}}j} = 1$.

According to (A) and (B), the only non-zero value in the sum in (C) must be $\Delta''_{(v'_{\widehat{I}})u'}$. Since Δ'' is witnessing $\widehat{I} \models (S_2, u)$, we have $(\widehat{I}, v'_{\widehat{I}}) \models (S_2, u')$. But $v'_{\widehat{I}}$ and p' only differ by state names, not by behaviors, so $(I', p') \models (S_2, u')$. This contradicts our assumption. Thus $v'\mathcal{R}u'$, and $\mathcal{R}$ is a weak refinement relation.

Finally, the hypothesis $[\![S_1]\!] \subseteq [\![S_2]\!]$ implies that $o_1\mathcal{R}o_2$ and further $S_1 \preceq S_2$. □

Thus, weak refinement and the implementation set inclusion coincide on the class of deterministic CMCs with at most one valuation in the initial state. Finally, Theorem 27 also holds for strong refinement, as the two refinements coincide for deterministic CMCs. Before any formal introduction of the result, we first introduce the following lemma that characterizes the shape of the witness matrix of the satisfaction relation for an implementation and a CMC in normal form.

Lemma 28. *Let $S = \langle\{1,\dots,k\}, o_S, \varphi, A, V_S\rangle$ be a deterministic CMC in single valuation normal form. Let $P = \langle\{1,\dots,n\}, o_P, M, A, V_P\rangle \in [\![S]\!]$ witnessed by a satisfaction relation $\mathcal{R}$. Let $p \in \{1,\dots,n\}$ and $u \in \{1,\dots,k\}$ such that $p\mathcal{R}u$, and let Δ be the associated correspondence matrix. We have*

$$\forall p' \in \{1,\dots,n\}.\, M_{pp'} \neq 0 \Rightarrow \left|\left\{u' \in \{1,\dots,k\} \mid \Delta_{p'u'} \neq 0\right\}\right| = 1.$$

Proof. Let $S = \langle\{1,\dots,k\}, o_S, \varphi, A, V_S\rangle$ be a deterministic *CMC* in single valuation normal form. Let $P = \langle\{1,\dots,n\}, o_P, M, A, V_P\rangle \in [\![S]\!]$ witnessed by a satisfaction relation $\mathcal{R}$. Let $p \in \{1,\dots,n\}$ and $u \in \{1,\dots,k\}$ such that $p\mathcal{R}u$, and let Δ be the associated correspondence matrix.

Suppose that there exist p', u' and u'' such that $M_{pp'} > 0$, $\Delta_{p'u'} > 0$ and $\Delta_{p'u''} > 0$ with $u' \neq u''$. Let $y = M\Delta$ be the probabilistic transition outgoing from u according to Δ. By construction, we have $y_{u'} > 0$ and $y_{u''} > 0$.

Moreover, since $\Delta_{p'u'} > 0$ and $\Delta_{p'u''} > 0$, it holds that $p'\mathcal{R}u'$ and $p'\mathcal{R}u''$. Because of the single valuation normal form of S, this implies that $V_S(u') = V_S(u'') = \{V_P(p')\}$.

Finally, there exist u, u' and $u'' \in \{1,\dots,k\}$ with $u' \neq u''$ and $y \in [0,1]^k$ such that $\varphi(u)(y) = 1$, $y_{u'} > 0$, $y_{u''} > 0$ and $V_S(u') = V_S(u'')$. This breaks the assumption that S is deterministic, which concludes the proof. □

According to the lemma, in any MC implementing a deterministic CMC, the probability of going to one implementation state is never distributed to more than one specification state. Otherwise the specification would be non-deterministic. We are now ready to state the theorem.

Theorem 29. *Let $S_1 = \langle\{1,\dots,k_1\}, o_1, \varphi_1, A, V_1\rangle$ and $S_2 = \langle\{1,\dots,k_2\}, o_2, \varphi_2, A, V_2\rangle$ be two deterministic CMCs in normal form. If there exists a weak refinement relation $\mathcal{R}$ such that $S_1\mathcal{R}S_2$, then $\mathcal{R}$ is also a strong refinement relation.*

Proof. Let $v \in \{1,\dots,k_1\}$ and $u \in \{1,\dots,k_2\}$ such that $v\mathcal{R}u$.

1. By hypothesis, $V_1(v) \subseteq V_2(u)$.
2. We know that for all $x \in [0,1]^{k_1}$ satisfying $\varphi_1(v)$, there exists a correspondence matrix Δ^x satisfying the axioms of weak refinement. We will build a correspondence matrix Δ^0 that will work for all x. Let $p \in \{1,\dots,k_1\}$.

   If for all $x \in [0,1]^{k_1}$, $\varphi_1(v)(x) \Rightarrow x_p = 0$, then let $\Delta^0_p = (0,\dots,0)$.

   Else, consider $x \in [0,1]^{k_1}$ such that $\varphi_1(v)(x)$ and $x_p \neq 0$. By hypothesis, there exists a correspondence matrix Δ^x associated to $v\mathcal{R}u$. Let $\Delta^0_p = \Delta^x_p$. By Lemma 28, there is a single $u' \in \{1,\dots,k_2\}$ such that $\Delta^x_{pu'} \neq 0$. Moreover, by definition of Δ^x, we know that $\sum_{r=1}^{k_2} \Delta^x_{pr} = 1$, thus $\Delta^x_{pu'} = 1$. So Δ^0 is a correspondence matrix.

   Suppose there exists $y \neq x \in [0,1]^{k_1}$ such that $\varphi_1(v)(y)$ and $y_p \neq 0$. Let Δ^y be the associated correspondence matrix. As for x, there exists a unique $u'' \in \{1,\dots,k_2\}$ such that $\Delta^y_{pu''} \neq 0$. Moreover $\Delta^y_{pu''} = 1$. Let $x' = x\Delta^x$ and $y' = y\Delta^y$. By definition, both $\varphi_2(u)(x')$ and $\varphi_2(u')(y')$ hold, $x'_{u'} \neq 0$ and $y'_{u''} \neq 0$. As $\Delta^x_{pu'} = \Delta^y_{pu''} = 1$, we have $V_2(u') \cap V_2(u'') \neq \emptyset$. By hypothesis, S_2 is deterministic, thus $u' = u''$.

   As a consequence, we have $\Delta^x_p = \Delta^y_p$, so $\forall z \in [0,1]^{k_1}$, $(\varphi_1(v)(z) \wedge (z_p \neq 0)) \Rightarrow \Delta^z_p = \Delta^0_p$, and we conclude that Δ^0 is uniquely defined.

   Finally, consider Δ^0 defined as above. Let $x \in [0,1]^{k_1}$ such that $\varphi_1(v)(x)$:
   - $x_i \neq 0 \Rightarrow \Delta^0_i = \Delta^x_i \Rightarrow \sum_{j=1}^{k_2} \Delta^0_{ij} = 1$;
   - $x\Delta^0 = x\Delta^x$, thus $\varphi_2(u)(x\Delta^0)$ holds.
   - If $\Delta^0_{v'u'} \neq 0$, then there exists $y \in [0,1]^{k_1}$ such that $\varphi_1(v)(y)$ and $\Delta^0_{v'u'} = \Delta^y_{v'u'}$, thus $v'\mathcal{R}u'$.

We conclude that $\mathcal{R}$ is a strong refinement relation. □

Let us summarize the relations between refinements for deterministic CMCs:

$$[\![S_1]\!] \subseteq [\![S_2]\!] \quad \text{iff} \quad S_1 \preceq S_2 \text{ iff } S_1 \text{ strongly refines } S_2, \tag{6}$$

if the CMCs are consistent and have at most one valuation in the initial state. The coincidence of the semantic refinement with coinductive refinements for CMCs is analogous to the case of trace inclusion refinement and simulation for deterministic transition systems.

The above results on completeness for deterministic specifications carry over to refinements of [11,16] for IMCs, which are special cases of our refinements. Completeness of these refinements was an open problem until now.

Discussion: a weaker notion of determinism. Our notion of determinism may appear overly strong. Indeed, it assumes that, from a given state i, one cannot reach two states u and v that share common sets of valuations. The assumption is made independently of the distributions used to reach the two states, i.e., it may be the case that there exists no distribution through which both u and v can be reached simultaneously.

A natural way to solve the problem would be to consider a weaker version of determinism. More precisely, we say that a CMC $S = \langle\{1, \ldots, k\}, o, \varphi, A, V\rangle$ is weakly deterministic if whenever there exists $x \in [0, 1]^k$ and states i, u, v such that $\varphi(i)(x)$ and $x_u > 0$ and $x_v > 0$, we have $V(u) \cap V(v) = \emptyset$. This version of determinism is strictly weaker than the one given in Definition 24. Indeed, only states that can be reached by the same distribution should have disjoint sets of valuations. Though this notion seems reasonable, as mentioned, one can show that there exist two weakly deterministic CMCs S_c and S_d such that $[\![S_c]\!] \subseteq [\![S_d]\!]$ but $S_c \not\preceq S_d$. Example of such CMCs are given in the second item of Proposition 12 on page 19. Hence working with this weaker, even if more natural, version of determinism does not close the gap between weak refinement and implementation set inclusion.

9. Polynomial CMCs

It is not surprising that CMCs are closed under both conjunction and parallel composition. Indeed, CMCs do not make any assumptions on constraint functions, even though there are many classes of constraints that are practically intractable. While this paper is mainly concerned with the development of the theoretical foundations for CMCs, we now briefly study classes of CMCs for which operations on constraints required by our algorithms can be managed quite efficiently.

A first candidate could be linear constraints, which is the obvious generalization of interval constraints. Unfortunately, linear constraint CMCs are not closed under parallel composition. Indeed, as we have seen in Section 6 the parallel composition of two linear constraints leads to a polynomial constraint. However, what is more interesting is that polynomial constraints *are* closed under both conjunction and parallel composition and that these operations do not increase the quantifier alternations since they only introduce existential quantifiers. Hence, one can claim that CMCs with polynomial constraints and only existential quantifiers are certainly the smallest extension of IMCs closed under all operations.

From the algorithmic point of view, working with polynomial constraints should not be seen as an obstacle. First, we observe that algorithms for conjunction and parallel composition do not require any complex operations on polynomials. The refinement algorithms (presented in Section 4.2) are polynomial in the number of states, and each iteration requires a quantifier elimination. This procedure is known to be double exponential in general, but there exist efficient single exponential algorithms [23,24] when quantifier alternations are fixed. Those algorithms are implemented in Maple [25]. The pruning operation is polynomial in the number of states, but each iteration also requires an exponential treatment as one has to decide whether the constraints have at least one solution. Again, such problems can be solved with efficient algorithms. Finally, determinizing a CMC can be performed with a procedure that is similar to the determinization procedure for finite-state automata. Such a procedure is naturally exponential in the number of states.

Remark 2. In Section 6, it was shown that, assuming independent sets of valuations, parallel composition is refined by conjunction. We have also observed that the conjunction or disjunction of two linear constraints remains linear, but that parallel composition may introduce polynomial constraints. From an implementation point of view it may thus be more efficient to work with linear constraints only. For doing so, one can simply approximate parallel composition with conjunction.

10. Relating CMCs to probabilistic automata

CMCs are a newcomer in a long series of probabilistic modeling languages and abstractions for them. Throughout the paper we have indicated that many of our results directly translate to simpler abstractions, like IMCs. We shall now further discuss this foundational aspect of CMCs, showing how they subsume a few established notions of refinement and parallel composition for probabilistic automata (and for process algebra based on them).

Below we write $\mathrm{Dist}(S)$ for the set of all probability distributions over a finite set S. Given two finite sets S and T and a probability distribution $\alpha \in \mathrm{Dist}(S \times T)$, we denote the marginal distribution over S as $\alpha_{s,T} = \sum_{t\in T} \alpha_{s,t}$, and similarly for T. We say that φ is a *non-deterministic distribution constraint* over set I if all solutions x of φ are point distributions; so $\exists i.\, x_i = 1$. Write $[{}^{\,i}_{S}]$ to denote a particular point distribution for which $[{}^{\,i}_{S}]_i = 1$. For example, in Fig. 11 constraints $\varphi_c(2)$ and $\varphi_d(1)$ are non-deterministic distribution constraints. The two-point distributions satisfying $\varphi_c(2)$ are $[{}^{\;3}_{1..4}]$ and $[{}^{\;4}_{1..4}]$.

Non-deterministic distribution constraints model a non-deterministic choice of an element from S. They will be used to encode non-determinism in CMCs.

A probabilistic automaton (PA for short) [17] is a tuple $\mathbb{S} = (S, Act, \rightarrow, s_1)$, where S is a finite set of states, $\rightarrow\, \subseteq S \times Act \times \mathrm{Dist}(S)$ is a finite transition relation and $s_1 \in S$ is the initial state.

If $\pi \in \mathrm{Dist}(S)$ and $\varrho \in \mathrm{Dist}(T)$, then $\pi \otimes \varrho$ denotes the unique independent product distribution such that $(\pi \otimes \varrho)_{st} = \pi_s \varrho_t$. This is consistent with our definition of $\otimes$ in Section 2, if π, ϱ and $\pi \otimes \varrho$ are interpreted as row vectors. Then the *derived combined transition relation* of $\mathbb{S}$ is given by $\rightarrow_c\, \in S \times Act \times \mathrm{Dist}(S)$ as follows. We say that $t \xrightarrow{a}_c \varrho$ iff ϱ is a convex linear combination of vectors from $\boldsymbol{\varrho} = \{\varrho_i \mid t \xrightarrow{a} \varrho_i\}$, so $\varrho = \boldsymbol{\varrho} \times \lambda$, where λ is a distribution vector $\lambda \in [0, 1]^{|\boldsymbol{\varrho}|}$. We interpret $\boldsymbol{\varrho}$ as a matrix, where ith column is a distribution ϱ_i.

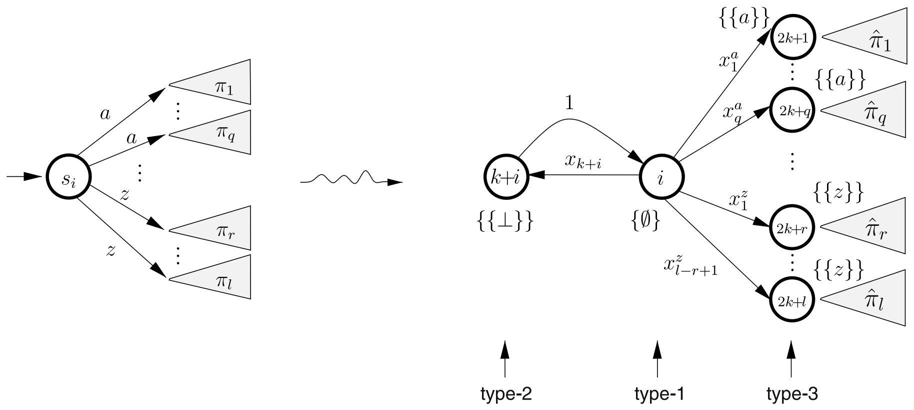


Fig. 18. Reducing a PA to CMC. There $\widehat{\pi}$ denotes a distribution constraint, which has a unique solution π. This is formalized below as $\widehat{\varphi}(2k + i')(x)$.

Consider two PA $\mathbb{S} = (S, Act, \rightarrow^S, s_0)$ and $\mathbb{T} = (T, Act, \rightarrow^T, t_0)$. For a binary relation $R \subseteq S \times T$ we define a derived relation $R^* \subseteq \text{Dist}(S) \times \text{Dist}(T)$ such that $\pi R^* \varrho$ iff there exists a distribution $\alpha \in \text{Dist}(S \times T)$ and (1) $\alpha_{q,T} = \pi_q$ for all $q \in S$, (2) $\alpha_{S,r} = \varrho_r$ for all $r \in T$ and (3) $\alpha_{s,t} \neq 0$ implies sRt.

Definition 30 (*Simulation [17]*)**.** A relation $R \subseteq S \times T$ is a *simulation* iff $(s, t) \in R$ implies that whenever $s \xrightarrow{a} \pi$ for a distribution π, then $t \xrightarrow{a} \varrho$ for distribution ϱ such that $\pi R^* \varrho$.

R is a *probabilistic* simulation iff $(s, t) \in R$ implies that if $s \xrightarrow{a} \pi$, then $t \xrightarrow{a}_C \varrho$ for some distribution ϱ, and $\pi R^* \varrho$.

Let $A \subseteq Act$ be the subset of actions on which $\mathbb{S}$ and $\mathbb{T}$ should synchronize. The *parallel composition* of $\mathbb{S}$ and $\mathbb{T}$ is a PA $\mathbb{S} \parallel \mathbb{T} = (S \times T, Act, \rightarrow, (s_0, t_0))$, where $\rightarrow$ is the largest transition relation such that $(s, t) \xrightarrow{a} \pi \otimes \varrho$ if:

$a \in A$ and $s \xrightarrow{a}{}^S \pi$ and $t \xrightarrow{a}{}^T \varrho$, or

$a \notin A$ and $s \xrightarrow{a}{}^S \pi$ and $\varrho = [^t_T]$, or

$a \notin A$ and $\pi = [^s_S]$ and $t \xrightarrow{a}{}^T \varrho$.

We now propose a linear encoding of PAs into CMCs, which reduces simulation and parallel composition of PAs to refinement and parallel composition of CMCs (see Fig. 18). Let $\mathbb{S} = (\{s_1, \ldots, s_k\}, Act, \rightarrow, s_0)$ be a PA. And let l be the number of reachable action–distribution pairs, so $\Omega_{\mathbb{S}} = \{(a_1, \pi_1), \ldots, (a_l, \pi_l)\} = \{(a, \pi) \mid \exists s \in S.\, s \xrightarrow{a} \pi\}$. The corresponding CMC is

$$\widehat{\mathbb{S}} = \langle \{1, \ldots, 2k + l\}, 1, \widehat{\varphi}, \text{Act} \cup \perp, \widehat{V} \rangle, \quad \text{where } \perp \notin Act.$$

$\widehat{\mathbb{S}}$ has three kinds of states. Type-1 states, $1 \ldots k$, correspond directly to states of $\mathbb{S}$. Distributions leaving these states model a non-deterministic choice. Type-2 states, $k + 1, \ldots, 2k$, model a possibility that a component remains idle in a state. Type-3 states, $2k + 1, \ldots, 2k + l$ model the actual distributions of $\mathbb{S}$. In the following we use i to range over states of Type-1 (so usually $1 \leq i \leq k$) and i' to range over action–distribution pairs (so usually $1 \leq i' \leq l$). Similarly for j.

$\widehat{V}$ assigns value $\{\emptyset\}$ to type-1 states and value $\{\{\perp\}\}$ to type-2 states. For type-3 states we assign actions of transitions in $\mathbb{S}$: $\widehat{V}(2k + i') = \{\{a_{i'}\}\}$ for $1 \leq i' \leq l$. The distribution constraints are as follows:

$\widehat{\varphi}(i)(x)$ iff i is type-1 and $x = [^{k+i}_{1..2k+l}]$ or $s_i \xrightarrow{a_{i'}} \pi_{i'} \wedge x = [^{2k+i'}_{1..2k+l}]$ for $1 \leq i' \leq l$.

$\widehat{\varphi}(k + i)(x)$ iff $k + i$ is type-2 and $x = [^{i}_{1..2k+l}]$.

$\widehat{\varphi}(2k + i')(x)$ iff $2k + i'$ is type-3 and $\forall j \in \{1, \ldots, k\}.\, x_j = \pi_{i'}(s_j)$.

We can now relate simulation of PAs to refinement of CMCs. We say that a constraint is a single-point constraint if it is only satisfied by a unique distribution. Observe that all constraints in the encoding presented above are non-deterministic distribution constraints or single-point constraints.

Lemma 31. *Let φ and ψ be single-point constraints. If for each $x \in [0, 1]^{k_1}$ such that $\varphi(x)$ holds, there exists a correspondence matrix $\Delta_x \in [0, 1]^{k_1 \times k_2}$ such that $\psi(x\Delta_x)$ holds, then there exists a correspondence matrix $\Delta \in [0, 1]^{k_1 \times k_2}$ such that for all $x \in [0, 1]^{k_1}$ we have that $\varphi(x) \implies \psi(x\Delta)$.*

The lemma holds trivially because there is only one distribution satisfying φ.

Lemma 32. *Let φ (respectively ψ) be a non-deterministic distribution constraint over $\{1, \ldots, k_1\}$ (respectively $\{1, \ldots, k_2\}$). Then if for each distribution vector x satisfying φ there exists a correspondence matrix $\Delta_x \in [0, 1]^{k_1 \times k_2}$ such that $\psi(x\Delta_x)$ holds, then there exists a correspondence matrix $\Delta \in [0, 1]^{k_1 \times k_2}$ such that for all $x \in [0, 1]^{k_1}$ we have that $\varphi(x) \implies \psi(x\Delta)$.*

Proof. Let x be such that $\varphi(x)$ holds (thus there exists $1 \leq i \leq k_1$ such that $x_i = 1$). There is a finite number of such vectors. Let x^i denote the one that has 1 on the ith position. Take Δ such that $\Delta_i = (\Delta_{x^i})_i$ (the witness from the lemma assumption) if x^i satisfies φ and $\Delta_i = 0^{k_2}$ otherwise.

Now for each x^i satisfying φ we have that $x^i\Delta = x^i\Delta_{x^i}$ and then $\varphi(x^i) \implies \psi(x^i\Delta_{x^i}) \iff \psi(x^i\Delta)$. $\square$

Corollary 33. *For any two probabilistic automata $\mathbb{S}$ and $\mathbb{T}$ we have that $\widehat{\mathbb{S}}$ strongly refines $\widehat{\mathbb{T}}$ iff $\widehat{\mathbb{S}}$ weakly refines $\widehat{\mathbb{T}}$.*

Lemma 34. *For any two probabilistic automata $\mathbb{S}$ and $\mathbb{T}$ such that $\mathbb{T}$ simulates $\mathbb{S}$ we have that $\widehat{\mathbb{S}}$ weakly refines $\widehat{\mathbb{T}}$.*

Proof (*Sketch*). Let $\mathcal{R} \subseteq S \times T$ be the relation witnessing the simulation of $\mathbb{S}$ by $\mathbb{T}$. Consider a relation $\mathcal{Q}$ as follows:

$$\begin{aligned}
\mathcal{Q}_1 &= \{(i,j) \mid i \in \{1,\dots,k_1\}, j \in \{1,\dots,k_2\}, (s_i,t_j) \in \mathcal{R}\}\\
\mathcal{Q}_2 &= \{(k_1+i, k_2+j) \mid i \in \{1,\dots,k_1\}, j \in \{1,\dots,k_2\}, (s_{i-k_1}, t_{j-k_2}) \in \mathcal{R}\}\\
\mathcal{Q}_3 &= \{(2k_1+i', 2k_2+j') \mid i' \in \{1,\dots,l_1\}, j' \in \{1,\dots,l_2\}, (a_i,\pi_i) \in \Omega_{\mathbb{S}}, (a_j,\varrho_j) \in \Omega_{\mathbb{T}}, a_i = a_j, (\pi_i,\varrho_i) \in \mathcal{R}^*\}\\
\mathcal{Q} &= \mathcal{Q}_1 \cup \mathcal{Q}_2 \cup \mathcal{Q}_3.
\end{aligned}$$

It is easy to show that Q is a weak refinement. First observe that valuations always match for pairs in Q. The valuation is empty for both S and T in Q_1, it is $\{\perp\}$ in Q_2, and $\{a_i\}$ in Q_3.

For a pair in $(i,j) \in Q_1$ a distribution vector x satisfying the constraint of S is always a point distribution. If $x_{k_1+i} = 1$, take $\Delta_{k_1+i,k_2+j} = 1$ and zero otherwise. If $x_{2k_1+i'} = 1$ take $\Delta_{2k_1+i',2k_2+j'} = 1$ and zero otherwise, where j' is such that $t_{j'} \xrightarrow{a_{i'}} \varrho_{j'}$ and $\pi_{i'}R^*\varrho_{j'}$.

For a pair $(k_1+i, k_2+j) \in Q_2$ take $\Delta_{ij} = 1$, and zero otherwise.

For a pair $(2k_1+i', 2k_2+j') \in Q_3$ take Δ such that for $(i,j) \in \{1,\dots,k_1\} \times \{1,\dots,k_2\}$ we have $\Delta_{ij} = \alpha_{ij}/x_i$, or zero if $x_i = 0$, where α is the distribution witnessing $\pi_{i'}R^*\varrho_{j'}$. $\square$

Lemma 35. *For any two probabilistic automata $\mathbb{S}$ and $\mathbb{T}$ such that $\widehat{\mathbb{S}}$ strongly refines $\widehat{\mathbb{T}}$ we have that $\mathbb{T}$ simulates $\mathbb{S}$.*

Proof (*Sketch*). Assume that $\widehat{\mathbb{S}}$ strongly refines $\widehat{\mathbb{T}}$ is witnessed by a relation $R \subseteq \{1,\dots,2k_1+l_1\} \times \{1,\dots,2k_2+l_2\}$. Show that a relation $\mathcal{Q} = \{(s_i,t_j) \in S \times T \mid (i,j) \in R, i \in \{1,\dots,k_1\}, j \in \{1,\dots,k_2\}\}$ is a simulation relation.

In the crucial point of the proof consider $\alpha_{s_it_j} = \Delta_{ij}\pi_{i'}(s_i)$, where $\pi_{i'}$ is a distribution being the only solution of a point constraint for state $i' \in \{2k_1,\dots,2k_2+l_1\}$. $\square$

Theorem 36 follows as a corollary from the above two lemmas and Corollary 33.

Theorem 36. *$\mathbb{T}$ simulates $\mathbb{S}$ iff $\widehat{\mathbb{S}}$ strongly refines $\widehat{\mathbb{T}}$.*

The same encoding is used to characterize parallel composition of PAs using parallel composition of CMCs.

We say that two CMCs S_1 and S_2 are isomorphic if there exists a bijection $f : \{1,\dots,k_1\} \to \{1,\dots,k_2\}$, such that $\varphi(v)$ is satisfied by $x \in [0,1]^{k_1}$ if and only if $\varphi(f(v))$ is satisfied by x.

Expression $S[{}^{a'_1}/_{a_1};\dots;{}^{a'_n}/_{a_n}]_{a_1,\dots,a_n\in Act}$ denotes a comprehended substitution, substituting a primed version of name a_i for each occurrence in a_i, for all actions in Act.

Theorem 37. *For two PAs $\mathbb{S}$ and $\mathbb{T}$ over the same set of actions Act and a synchronizing set $A \subseteq Act$ we have that $\widehat{\mathbb{S} \parallel \mathbb{T}}$ is isomorphic to*

$$((\widehat{\mathbb{S}} \parallel \widehat{\mathbb{T}}[{}^{a'}/_{a}]_{a\in Act\cup\perp}) \wedge \mathsf{S}_A)\,[{}^{a}/_{(a,a')};\ {}^{a}/_{(a,\perp')};\ {}^{a}/_{(\perp,a')}]_{a\in Act},$$

where S_A is a synchronizer over $Act \cup \perp \times Act' \cup \perp'$ defined by

$$(\forall a \in A.\, a \iff a') \wedge (\forall a \notin A.\, (a \implies \perp') \wedge (a' \implies \perp)).$$

Proof. Let $\mathbb{S} = (\{s_1,\dots,s_{k_1}\}, Act, \to^S, s_1)$, $\mathbb{T} = (\{t_1,\dots,t_{k_2}\}, Act, \to^T, t_1)$, and $A \subseteq Act$. Consider $\mathbb{S} \parallel \mathbb{T} = (\{(s_1,t_1),(s_1,t_2),\dots,(s_{k_1},t_{k_2})\}, Act, \to, (s_1,t_1))$ defined in the usual way.

We now construct $\widehat{\mathbb{S} \parallel \mathbb{T}} = (\{1,\dots,2k_1k_2+l\}, 1, \widehat{\varphi}, Act \cup \perp, \widehat{V}\})$ in the usual way, where l is the number of reachable action–distribution pairs.

Consider $\widehat{\mathbb{S}} \parallel \widehat{\mathbb{T}}[{}^{a'}/_{a}]_{a\in Act\cup\perp} = \langle\{1,\dots,2k_1+l_1\} \times \{1,\dots,2k_2+l_2\}, (1,1), \varphi, Act \cup Act' \cup \perp \cup \perp', V\rangle$, where l_1 and l_2 are the number of reachable action–distribution pairs for $\mathbb{S}$ and $\mathbb{T}$, respectively. Conjoining with S_A allows exactly those pairs of actions that are allowed in the parallel composition of probabilistic automata.

Finally we apply the renaming $[{}^{a}/_{(a,a')};\ {}^{a}/_{(a,\perp')};\ {}^{a}/_{(\perp,a')}]_{a\in Act}$, and obtain $((\widehat{\mathbb{S}} \parallel \widehat{\mathbb{T}}[{}^{a'}/_{a}]_{a\in Act\cup\perp}) \wedge \mathsf{S}_A)[{}^{a}/_{(a,a')};\ {}^{a}/_{(a,\perp')};\ {}^{a}/_{(\perp,a')}]_{a\in Act}$.

The bijection will be taken as the mapping that takes a state in $\widehat{\mathbb{S} \parallel \mathbb{T}}$ to the equivalent state in $((\widehat{\mathbb{S}} \parallel \widehat{\mathbb{T}}[{}^{a'}/_{a}]_{a\in Act\cup\perp}) \wedge \mathsf{S}_A)\,[{}^{a}/_{(a,a')};\ {}^{a}/_{(a,\perp')};\ {}^{a}/_{(\perp,a')}]_{a\in Act}$. The bijection f is defined, for states allowed by the parallel composition of PAs, as follows:

- $i \in \{1,\dots,k_1k_2\}$ is mapped into $\{(1,1),\dots,(k_1,k_2)\}$ by $i \mapsto (((i-1)\,\mathrm{div}\,k_2)+1, ((i-1)\bmod k_2)+1)$,
- $i \in \{k_1k_2+1,\dots,2k_1k_2\}$ is mapped into $\{(k_1+1,k_2+1),\dots,(2k_1,2k_2)\}$ by $i \mapsto (((i-1)\,\mathrm{div}\,k_2)+1, ((i-1)\bmod k_2)+1+k_2)$, and

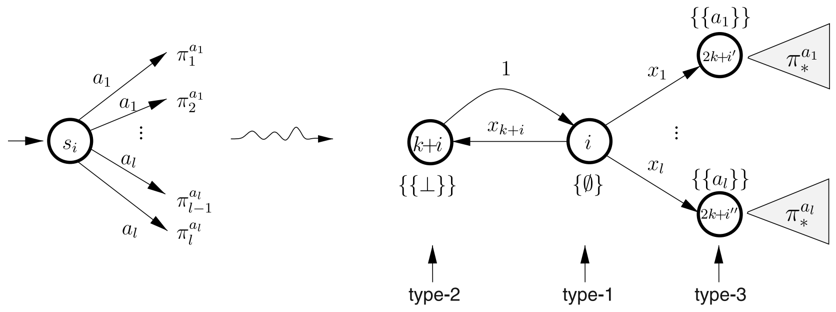


Fig. 19. An attempt to visualize the second encoding. π_*^a denotes a constraint expressing a probability vector that is a linear combination of all probability distributions labeled by a. Below this is formalized as $\check{\varphi}(2k+i')(x)$.

- $i \in \{2k_1k_2+1, \ldots, 2k_1k_2+l\}$ is mapped injectively into $\{k_1+1, \ldots, 2k_1\} \times \{2k_2+1, \ldots, 2k_2+l_2\} \cup \{2k_1+1, \ldots, 2k_1+l_1\} \times \{k_2+1, \ldots, 2k_2\} \cup \{2k_1+1, \ldots, 2k_1+l_1\} \times \{2k_2+1, \ldots, 2k_2+l_2\}$. This is done such that, $f(2k_1k_2+p) = (2k_1+q, 2k_2+r)$, if both $2k_1k_2+p$ and $(2k_1+q, 2k_2+r)$ are labeled with an $a \in A$ and there exists $s \in \{s_1, \ldots, s_{k_1}\}$ and $t \in \{t_1, \ldots, t_{k_2}\}$, such that $s \xrightarrow{a}{}^S \pi$ and $t \xrightarrow{a}{}^T \varrho$ and $(s,t) \xrightarrow{a} \pi \otimes \varrho$ and the constraint functions of $2k_1+q$ and $2k_2+r$ are satisfied by π and ϱ, respectively and the constraint function of $2k_1k_2+p$ is satisfied by $\pi \otimes \varrho$.
  Similarly, $f(2k_1k_2+p) = (k_1+q, 2k_2+r)$, if both $2k_1k_2+p$ and $(k_1+q, 2k_2+r)$ are labeled with an $a \notin A$ and there exists $s \in \{s_1, \ldots, s_{k_1}\}$ and $t \in \{t_1, \ldots, t_{k_2}\}$, such that $t \xrightarrow{a}{}^T \varrho$ and $\pi = [{}_{\{s_1,\ldots,s_k\}}^{\;\;s}]$ and $(s,t) \xrightarrow{a} \pi \otimes \varrho$.

From the above, is it clear that for type-3 states i, i and $f(i)$ will have equivalent constraint functions. For a type-1 state i, a distribution giving probability 1 to $i + k_1k_2$ is allowed. In $f(i)$ the same distribution is allowed, since, in the constraints of $\widehat{\mathbb{S}}$ and $\widehat{\mathbb{T}}$, distributions allowing $\mathbb{S}$ and $\mathbb{T}$ to idle, are allowed. Same argument holds for type-2 states.

It is then clear, that for a state v of $\widehat{\mathbb{S} \parallel \mathbb{T}}$, v and $f(v)$ have equivalent constraint functions. □

Interestingly, the precongruence property for parallel composition of PAs is obtained as a corollary of the above two reduction theorems and Theorem 17.

Another, very similar, but slightly more complicated, encoding exists, for which weak refinement coincides with *probabilistic* simulation. Consider a PA $\mathbb{S} = (S, Act, \to, s_1)$, where $S = \{s_1, \ldots, s_k\}$. Let $\{(s^1, a_1), \ldots, (s^l, a_l)\} = \{(s,a) \mid s \in S \wedge a \in Act\}$. The corresponding CMC is

$$\check{\mathbb{S}} = (\{1, \ldots, 2k+l\}, 1, \check{\varphi}, Act \cup \bot, \check{V}\}),$$

where $\bot$ is a fresh symbol not in Act. We have three types of states (see Fig. 19). Type-1 states, $\{1, \ldots, k\}$, correspond directly to states $\{s_1, \ldots, s_k\}$—their distribution constraints encode the non-deterministic choice of action. Type-2 states, $\{k+1, \ldots, 2k\}$, represent the ability of a state to be idle. We will use them in parallel composition. Type-3 states, $\{2k+1, \ldots, 2k+l\}$, encode choice of a probability distribution as a linear combination of distributions allowed by the automaton.

The valuation functions are given by:

$$\begin{array}{ll} \check{V}(i) = \{\emptyset\} & \text{for } 1 \le i \le k \\ \check{V}(k+i) = \{\{\bot\}\} & \text{for } 1 \le i \le k \\ \check{V}(2k+i') = \{\{a_{i'}\}\} & \text{for } 1 \le i' \le l \end{array}$$

and

$$\begin{array}{ll} \check{\varphi}(i)(x) \text{ is } x_{k+i} = 1 \text{ or } \exists 1 \le i' \le l.\ x_{2k+i'} = 1 \wedge s^{i'} = s_i & \text{for } 1 \le i \le k \text{ (type-1)} \\ \check{\varphi}(k+i)(x) \text{ is } x_i = 1 & \text{for } 1 \le i \le k \text{ (type-2)} \\ \check{\varphi}(2k+i')(x) \text{ is } \exists \lambda \in \text{Dist}(1, \ldots, |\boldsymbol{\pi}|).\, x = \boldsymbol{\pi}\lambda & \text{for } 1 \le i' \le l \text{ (type-3)}, \end{array}$$

where $\boldsymbol{\pi} = \{\pi \mid s^j \xrightarrow{a_j} \pi\}$. Technically speaking $\boldsymbol{\pi}$ is a matrix, whose columns are distributions π. We write $|\boldsymbol{\pi}|$ for the number of columns in $\boldsymbol{\pi}$. Additionally x is implicitly required to be a probability distribution over $\{1, \ldots, 2k+l\}$.

Observe that $\widehat{\mathbb{S}}$ is only polynomially larger than $\mathbb{S}$.

Lemma 38 (*Soundness*). *For any two probabilistic automata $\mathbb{S}$ and $\mathbb{T}$ such that $\check{\mathbb{S}}$ weakly refines $\check{\mathbb{T}}$, we have that $\mathbb{T}$ probabilistically simulates $\mathbb{S}$.*

Proof. Let $\mathbb{S} = (S, Act, \to^S, s_1)$ and $\mathbb{T} = (T, Act, \to^T, t_1)$, with $S = \{s_1, \ldots, s_{k_1}\}$ and $T = \{t_1, \ldots, t_{k_2}\}$. In the proof we write $\check{\varphi}$ to refer to the constraint function of $\check{\mathbb{S}}$, and $\check{\varrho}$ to refer to the constraint function of $\check{\mathbb{T}}$. Also l_1 and l_2 are used to refer to the number of combinations of state actions of respectively $\check{\mathbb{S}}$ and $\check{\mathbb{T}}$. Finally q_i and r_j are used to range over states in S (respectively in T), when s_i and t_j are bound to some concrete value.

Let $\mathcal{R} \in \{1, \ldots, 2k_1 + l_1\} \times \{1, \ldots, 2k_2 + l_2\}$ be a weak refinement relation between $\check{\mathbb{S}}$ and $\check{\mathbb{T}}$, witnessing the assumption of the lemma. The proof proceeds by showing that

$$\mathcal{Q} = \{(s_i, t_j) \mid (i, j) \in \mathcal{R} \wedge 1 \leq i \leq k_1 \wedge 1 \leq j \leq k_2\}$$

is a probabilistic simulation relation between $\mathbb{S}$ and $\mathbb{T}$.

We apply the usual coinductive proof technique. Take $(s_i, t_j) \in \mathcal{Q}$. Let $\pi \in \text{Dist}(S)$ be such that $s_i \xrightarrow{a} \pi$, and $(s^{i'}, a_{i'}) = (s_i, a)$.[2]

By construction of the encoding we know that any probability distribution x satisfying $\varphi(i)(x)$ is a point distribution, and x such that $x_{2k+i'} = 1$ is possible. So consider such a distribution x. Since $(i, j) \in \mathcal{R}$ we know that there exists a correspondence matrix $\Delta \in [0, 1]^{(2k_1+l_1)\times(2k_2+l_2)}$ such that $\psi(j)(x\Delta)$ holds. Moreover $x\Delta$ must be a point distribution by construction of the encoding. So $(x\Delta)_{2k_2+j'} = 1$ for some $1 \leq j' \leq l_2$. And, by refinement again, we get that valuation functions for $2k_1 + i'$ and for $2k_2 + j'$ both return $\{\{a\}\}$ and that $(2k_1 + i', 2k_2 + j') \in \mathcal{R}$.

But $\check{\mathbb{T}}$ is also constructed using the encoding, so it necessarily is that $t_j \xrightarrow{a} \varrho$ for some $\varrho \in \text{Dist}(T)$.

Observe that $\varphi(2k_1 + i')(\pi)$ holds, because π is always a convex linear combination of a set of vectors containing it. Since $(2k_1 + i', 2k_2 + j') \in \mathcal{R}$, there exists a correspondence matrix $\Delta' \in [0, 1]^{(2k_1+l_1)\times(2k_2+l_2)}$ such that $\psi(2k_2 + j')(\pi\Delta')$ holds. The latter implies that $\pi\Delta'$ is a linear combination of vectors in $\boldsymbol{\varrho} = \{\varrho \mid t_j \xrightarrow{a} \varrho\}$.

It remains to show that $\pi\mathcal{R}^*(\pi\Delta')$. Take $\alpha_{q_i,q_j} = \pi_i\Delta'_{ij}$. We first argue that $\alpha \in \text{Dist}(S \times T)$. Since each row of a correspondence matrix sums up to 1, we have that $\pi_i\Delta'_{ij} \in [0, 1]$ for all i, j. Also $\sum_{i=1}^{k_1}\sum_{j=1}^{k2} \pi_i\Delta'_{ij} = \sum_{i=1}^{k_1} \pi_i = 1$.

Consider $\alpha_{q_i,T} = \sum_{j=1}^{k_2} \alpha_{q_i,t_j} = \sum_{j=1}^{k_2} \pi_i\Delta'_{ij} = \pi_i \sum_{j=1}^{k_2} \Delta'_{ij} = \pi_i$ as required by $\pi\mathcal{R}^*(\pi\Delta')$.

Now consider $\alpha_{S,r_j} = \sum_{i=1}^{k_1} \alpha_{s_i,r_j} = \sum_{i=1}^{k_1} \pi_i\Delta'_{ij} = (\pi\Delta')_j$ as required by $\pi\mathcal{R}^*(\pi\Delta')$.

Now if $\alpha_{q_i,r_j} \neq 0$, then $\Delta'_{ij} \neq 0$, which in turn with refinement of $2k_2 + j'$ by $2k_1 + i'$ implies that $(i, j) \in \mathcal{R}$, and furthermore $(s_i, s_j) \in \mathcal{Q}$ by construction, as required by $\pi\mathcal{R}^*(\pi\Delta')$. This finishes the proof. □

Lemma 39 (*Completeness*). *For any two probabilistic automata $\mathbb{S}$ and $\mathbb{T}$ such that $\mathbb{T}$ probabilistically simulates $\mathbb{S}$, we have that $\check{\mathbb{S}}$ weakly refines $\check{\mathbb{T}}$.*

Proof. Let $\mathbb{S} = (S, Act, \to^S, s_1)$ and $\mathbb{T} = (T, Act, \to^T, t_1)$, with $S = \{s_1, \ldots, s_{k_1}\}$ and $T = \{t_1, \ldots, t_{k_2}\}$. Let $\mathcal{Q} \subseteq S \times T$ be the probabilistic simulation relation between $\mathbb{S}$ and $\mathbb{T}$, witnessing the assumption of the lemma.

The proof proceeds by showing that a relation $\mathcal{R} \subseteq \{1, \ldots, 2k_1 + l_1\} \times \{1, \ldots, 2k_2 + l_2\}$ is a weak refinement relation between $\check{\mathbb{S}}$ and $\check{\mathbb{T}}$.

Take the following candidate for $\mathcal{R}$:

$$\mathcal{R}_1 = \{(i, j) \mid (s_i, t_j) \in \mathcal{Q}\}$$
$$\mathcal{R}_2 = \{(k_1 + i, k_2 + j) \mid (s_i, t_j) \in \mathcal{Q}\}$$
$$\mathcal{R}_3 = \{(2k_1 + i', 2k_2 + j') \mid (s_i, t_j) \in R \wedge s_i = s^{i'} \wedge t_j = t^{j'}\}$$
$$\mathcal{R} = \mathcal{R}_1 \cup \mathcal{R}_2 \cup \mathcal{R}_3.$$

We apply the usual coinductive proof technique.

Case 1. Take $(i, j) \in \mathcal{R}_1$ and x satisfying $\varphi(i)(x)$. We know that x can only be a point distribution. If $x_{k_1+i} = 1$, then we take Δ such that $\Delta_{k_1+i,k_2+j} = 1$ (and Δ is zero for all other elements). Clearly Δ is a correspondence matrix. Moreover $x\Delta$ is a point distribution with 1 in the $(k_2 + j)$th position, so $\psi(j)(x\Delta)$ holds by construction of the encoding (see first case in encoding of constraints). Also $(k_1 + i, k_2 + j) \in \mathcal{R}_2$ since $(s_i, t_j) \in \mathcal{Q}$.

If $x_{2k_1+i'} = 1$, then it means that $s_i \xrightarrow{\check{V}(i)} \pi$ for some π and action $\check{V}(i)$. But then, since $(s_i, t_j) \in \mathcal{Q}$, it is possible that $t_j \xrightarrow{\check{V}(i)}_c \varrho$, for some distribution ϱ. Let j' be such that $t_j = t^{j'}$ and $a_{j'} = \check{V}(i)$. Take a correspondence matrix Δ such that $\Delta_{2k_1+i',2k_2+j'} = 1$ (and Δ is zero for all other elements). We have that $x\Delta$ is a point distribution with 1 in the $(2k_2 + j')$th position, so $\psi(j)(x\Delta)$ holds by construction of encoding resulting in j (see first case in encoding of constraints). Also $(2k_1 + i', 2k_2 + j') \in \mathcal{R}_3 \subseteq \mathcal{R}$ by definition of $\mathcal{R}_3$.

Case 2. Take $(k_1 + i, k_2 + j) \in \mathcal{R}_2$. The argument is almost identical to the first sub-case in Case 1. We omit it here.

Case 3. Take $(2k_1 + i', 2k_2 + j') \in \mathcal{R}_3$ and x satisfying $\varphi(2k_1 + i')(x)$. Let $s_i = s^{i'}$ and $t_j = t^{j'}$. By $\mathcal{R}_3$ we know that $(s_i, t_j) \in \mathcal{Q}$. By construction of the encoding $s_i \xrightarrow{\check{V}(2k_1+i')} x$ and furthermore $t_j \xrightarrow{\check{V}(2k_1+i')}_c \varrho$, where $\varrho = \boldsymbol{\varrho}\lambda$ for some probability distribution $\lambda \in \text{Dist}(1, \ldots, |\boldsymbol{\varrho}|)$. Clearly $\psi(2k_2 + j')(\varrho) = 1$. It remains to check that π can be correspondence to ϱ.

To this end consider a correspondence matrix Δ such that

$$\Delta_{ij} = \begin{cases} \alpha_{s_i,t_j}/x_i & \text{if } x_i \neq 0 \text{ and } i \leq k_1, j \leq k_2 \\ 0 & \text{otherwise.} \end{cases}$$

2 The equality binds i' to be the index of (s_i, a) on the list of state–action pairs in the encoding of $\mathbb{S}$.

Now $(x\Delta)_j = \sum_{i=1}^{2k_1+l_1} x_i\Delta_{ij} = \sum_{i=1}^{k_1} x_i\alpha_{s_i,t_j}/x_i = \sum_{i=1}^{k_1} \alpha_{s_i,t_j} = \alpha_{S,t_j} = \varrho_j$ by $xR^*\varrho$ (this discussion only holds for $j \leq k_2$, but the remaining elements are zero, which is easy to argue for. Also somewhat sloppily we ignored the possibility of division by zero — indeed it cannot happen since for $x_i = 0$ we said that Δ_{ij} is simply zero). Effectively $x\Delta = \varrho$, so it satisfies $\psi(2k_2+j')$. Valuations obviously match.

Moreover if $\Delta_{ij} \neq 0$, then $\alpha_{s_i,t_j} \neq 0$ and $(s_i, t_j) \in \mathcal{Q}$ and then $(i, j) \in \mathcal{R}_1 \subseteq \mathcal{R}$, which finishes the proof. □

Theorem 40 is a corollary from the above two lemmas.

Theorem 40. $\mathbb{T}$ *probabilistically simulates* $\mathbb{S}$ *iff* $\check{\mathbb{S}}$ *weakly refines* $\check{\mathbb{T}}$.

Similarly, we obtain a precongruence with probabilistic simulation using a suitable encoding—a good example how CMCs can be used to study properties of simpler languages in a generic way.

11. Related work and concluding remarks

We have presented CMCs—a new model for representing a possibly infinite family of MCs. Unlike the previous attempts [11,16], our model is closed under many design operations, including composition, conjunction, determinization and normalization. We have studied these operations as well as several classical compositional reasoning properties, showing that, among others, the CMC specification theory is equipped with a complete refinement relation (for deterministic specifications), which naturally interacts with parallel composition, synchronization and conjunction. We have also demonstrated how our framework can be used to obtain properties for less expressive languages, by using reductions. In particular, we have exemplified this for probabilistic automata with simulation and probabilistic simulation of Segala.

Two recent contributions [16,29] are related to our work. Fecher et al. [16] propose a model checking procedure for PCTL [30] and Interval Markov Chains (other procedures recently appeared in [22,31]), which is based on weak refinement. However, our objective is not to use CMCs within a model checking procedure for probabilistic systems, but rather to benefit from it as a specification theory.

Very recently Katoen et al. [29] have extended Fecher's work to *Interactive* Markov Chains, a model for performance evaluation [32,33]. Their abstraction uses the continuous-time version of IMCs [34] augmented with may and must transitions, very much in the spirit of [2]. Parallel composition is defined and studied for this abstraction, however conjunction has been studied neither in [16] nor in [29].

Over the years process algebraic frameworks have been proposed for describing and analyzing probabilistic systems based on Markov Chains (MCs) and Markov Decision Processes [20,35,36]. Also a variety of probabilistic logics have been developed for expressing properties of such systems, e.g., PCTL [12]. Both traditions support refinement between specifications using various notions of probabilistic simulation [11,16] and, respectively, logical entailment [37]. Whereas the process algebraic approach favors parallel composition, the logical approach favors conjunction. Neither of the two supports *both* conjunction and parallel composition.

In mathematics the abstraction of Markov set-chains [38] lies very close to IMCs. It has been, for instance, used to approximate dynamics of hybrid systems [39]. The latter defines the intervals on the transition probabilities, while the former uses matrix intervals in the transition matrix space, which allows reasoning about the abstraction using linear algebra. Technically a Markov set-chain is an explicit enumeration of all the implementations of an IMC. While these works are clearly related to ours, we shall observe that like IMCs, these models are not closed under conjunction/composition.

In controller synthesis a notion of *Constrained Markov Decision Processes* (CMDPs) has been introduced. The similarity of name to CMCs is purely coincidental. In particular CMDPs are not a generalization/abstraction of CMCs. CMDPs, as described by Altman [40], are Markov Decision Processes annotated with several cost functions. They are used to synthesize probabilistic schedulers that optimize one cost function under a constraint over the other functions. Thus they are not a specification theory or an abstraction in the same sense as CMCs are.

As a future work, it would be of interest to design, implement and evaluate efficient algorithms for procedures outlined in this paper. We would also like to define a quotient relation for CMCs, presumably building on results presented in [41]. The quotienting operation is of particular importance for component reuse [4,42–45]. One could also investigate the applicability of our approach in model checking procedures, in the same style as Fecher and coauthors have used IMCs for model checking PCTL [16]. Another promising direction would be to mix our results with those we recently obtained for timed specifications [46–48], hence leading to the first theory for specification of timed probabilistic systems [49]. We should also investigate more quantitative versions of the refinement operation like this was done for contracts in [50]. Finally, it would be interesting to extend our composition operation by considering products of dependent probability distributions in the spirit of [51].

Acknowledgments

This work was supported by the European COMBEST project no. 215543, by VKR Centre of Excellence MT-LAB, and by an "Action de Recherche Collaborative" ARC (TP)I.

References

[1] T.A. Henzinger, J. Sifakis, The embedded systems design challenge, in: Formal Methods, FM, in: Lecture Notes in Computer Science, vol. 4085, Springer, 2006, pp. 1–15.
[2] K.G. Larsen, Modal specifications, in: Automatic Verification Methods for Finite State Systems, AVMS, in: Lecture Notes in Computer Science, vol. 407, Springer, 1989, pp. 232–246.
[3] K. G. Larsen, U. Nyman, A. Wąsowski, On modal refinement and consistency, in: Concurrency Theory, CONCUR, Springer, 2007, pp. 105–119.
[4] J.-B. Raclet, E. Badouel, A. Benveniste, B. Caillaud, A. Legay, R. Passerone, Modal interfaces: unifying interface automata and modal specifications, in: Embedded Software, EMSOFT, ACM, 2009, pp. 87–96.
[5] K.G. Larsen, U. Nyman, A. Wąsowski, Modal I/O automata for interface and product line theories, in: European Symposium on Programming, ESOP, in: Lecture Notes in Computer Science, Springer, 2007, pp. 64–79.
[6] L. de Alfaro, T.A. Henzinger, Interface automata, in: Foundations of Software Engineering, FSE, ACM Press, 2001, pp. 109–120.
[7] L. Doyen, T.A. Henzinger, B. Jobstmann, T. Petrov, Interface theories with component reuse, in: L. de Alfaro, J. Palsberg (Eds.), Embedded Software, EMSOFT, ACM Press, 2008, pp. 79–88.
[8] A. Chakrabarti, L. de Alfaro, T.A. Henzinger, F.Y.C. Mang, Computer Aided Verification, CAV, in: Lecture Notes in Computer Science, vol. 2404, 2002, pp. 414–427.
[9] L. de Alfaro, L.D. da Silva, M. Faella, A. Legay, P. Roy, M. Sorea, Sociable interfaces, in: Frontiers of Combining Systems, FroCos, in: Lecture Notes in Computer Science, vol. 3717, Springer, 2005, pp. 81–105.
[10] B.T. Adler, L. de Alfaro, L.D. da Silva, M. Faella, A. Legay, V. Raman, P. Roy, Ticc: a tool for interface compatibility and composition, in: Computer Aided Verification, CAV, in: Lecture Notes in Computer Science, vol. 4144, Springer, 2006, pp. 59–62.
[11] B. Jonsson, K.G. Larsen, Specification and refinement of probabilistic processes, in: Logic in Computer Science, LICS, IEEE Computer, 1991, pp. 266–277.
[12] H. Hansson, B. Jonsson, A logic for reasoning about time and reliability, Form. Asp. Comput. 6 (5) (1994) 512–535.
[13] T. Brázdil, V. Forejt, J. Kretínský, A. Kucera, The satisfiability problem for probabilistic ctl, in: Proceedings of the Twenty-Third Annual IEEE Symposium on Logic in Computer Science, LICS 2008, 24–27 June 2008, IEEE Computer Society, Pittsburgh, PA, USA, 2008, pp. 391–402.
[14] A. David, K.G. Larsen, A. Legay, U. Nyman, A. Wasowski, Methodologies for specification of real-time systems using timed i/o automata, in: F.S. de Boer, M.M. Bonsangue, S. Hallerstede, M. Leuschel (Eds.), FMCO, in: Lecture Notes in Computer Science, vol. 6286, Springer, 2009, pp. 290–310.
[15] L. de Alfaro, T.A. Henzinger, Interface-based design, in: Engineering Theories of Software-intensive Systems, in: NATO Science Series: Mathematics, Physics, and Chemistry, vol. 195, Springer, 2005, pp. 83–104.
[16] H. Fecher, M. Leucker, V. Wolf, Don't Know in probabilistic systems, in: SPIN, in: Lecture Notes in Computer Science, vol. 3925, Springer, 2006, pp. 71–88.
[17] R. Segala, N. Lynch, Probabilistic simulations for probabilistic processes, in: Concurrency Theory CONCUR, in: Lecture Notes in Computer Science, vol. 836, springer, 1994, pp. 481–496.
[18] H. Hansson, B. Jonsson, A calculus for communicating systems with time and probabitilies, in: IEEE Real-Time Systems Symposium, 1990, pp. 278–287.
[19] B. Jonsson, K. Larsen, W. Yi, Probabilistic extensions of process algebras, in: Handbook of Process Algebra, Elsevier, 2001, pp. 681–710.
[20] H. Hermanns, Interactive Markov chains: and the quest for quantified quality, Springer-Verlag, Berlin, Heidelberg, 2002.
[21] K. Sen, M. Viswanathan, G. Agha, Model-checking Markov chains in the presence of uncertainties, in: Tools and Algorithms for the Construction and Analysis of Systems, TACAS, in: Lecture Notes in Computer Science, vol. 3920, Springer, 2006, pp. 394–410.
[22] K. Chatterjee, K. Sen, T.A. Henzinger, Model-checking omega-regular properties of interval Markov chains, in: Foundations of Software Science and Computation Structures, FoSSaCS, in: Lecture Notes in Computer Science, vol. 4962, Springer, 2008, pp. 302–317.
[23] C.W. Brown, Simple cad construction and its applications, J. Symbolic Comput. 31 (5) (2001) 521–547.
[24] C.W. Brown, J.H. Davenport, The complexity of quantifier elimination and cylindrical algeraic decomposition, in: Symbolic and Algebraic Computation SSAC, Waterloo, ON, Canada, 2007, pp. 54–60.
[25] H. Yanami, H. Anai, SyNRAC: a Maple toolbox for solving real algebraic constraints, ACM Commun. Comput. Algebra 41 (3) (2007) 112–113.
[26] S. Basu, New results on quantifier elimination over real closed fields and applications to constraint databases, J. ACM 46 (4) (1999) 537–555.
[27] B. Caillaud, B. Delahaye, K.G. Larsen, A. Legay, M.L. Pedersen, A. Wąsowski, Decision problems for interval markov chains, Res. Rep. (2010). URL http://www.cs.aau.dk/~mikkelp/doc/IMCpaper.pdf.
[28] N. Beneš, J. Křetínský, K. Larsen, J. Srba, On determinism in modal transition systems, Theoret. Comput. Sci. 410 (41) (2009) 4026–4043.
[29] J. Katoen, D. Klink, M.R. Neuhäußer, Compositional abstraction for stochastic systems, in: Formal Modelling and Analysis of Timed Systems, FORMATS, in: Lecture Notes in Computer Science, vol. 5813, Springer, 2009, pp. 195–211.
[30] F. Ciesinski, M. Größer, On probabilistic computation tree logic, in: Validation of Stochastic Systems, VSS, in: Lecture Notes in Computer Science, vol. 2925, Springer, 2004, pp. 147–188.
[31] S. Haddad, N. Pekergin, Using stochastic comparison for efficient model checking of uncertain Markov chains, in: Quantitative Evaluation of SysTems, QEST, IEEE Computer Society Press, 2009, pp. 177–186.
[32] H. Hermanns, U. Herzog, J. Katoen, Process algebra for performance evaluation, Theoret. Comput. Sci. 274 (1–2) (2002) 43–87.
[33] J. Hillston, A Compositional Approach to Performance Modelling, Cambridge University Press, 1996.
[34] J. Katoen, D. Klink, M. Leucker, V. Wolf, Three-valued abstraction for continuous-time Markov chains, in: Computer Aided Verification, CAV, in: Lecture Notes in Computer Science, vol. 4590, Springer, 2007, pp. 311–324.
[35] S. Andova, Process algebra with probabilistic choice, in: AMAST Workshop on Real-Time and Probabilistic Systems, ARTS, in: Lecture Notes in Computer Science, vol. 1601, Springer, 1999, pp. 111–129.
[36] N. López, M. Núñez, An overview of probabilistic process algebras and their equivalences, in: Validation of Stochastic Systems, VSS, in: Lecture Notes in Computer Science, vol. 2925, Springer, 2004, pp. 89–123.
[37] H. Hermanns, B. Wachter, L. Zhang, Probabilistic CEGAR, in: Computer Aided Verification, CAV, in: Lecture Notes in Computer Science, vol. 5123, Springer, 2008, pp. 162–175.
[38] H.J. Hartfield, Markov Set-Chains, in: Lecture Notes in Mathematics, vol. 1695, Springer Verlag, 1998.
[39] A. Abate, A. D'Innocenzo, M.D.D. Benedetto, S.S. Sastry, Markov set-chains as abstractions of stochastic hybrid sytems, in: M. Egerstedt, B. Mishra (Eds.), Proceedings of the 11th International Workshop on Hybrid Systems: Computation and Control, in: Lecture Notes in Computer Science, vol. 4981, Springer, 2008, pp. 1–15.
[40] E. Altman, Constrained Markov Decision Processes, Chapman & Hall/CRC, 1999.
[41] K.G. Larsen, A. Skou, Compositional verification of probabilistic processes, in: Concurrency Theory, CONCUR, in: Lecture Notes in Computer Science, vol. 630, Springer, 1992, pp. 456–471.
[42] J.-B. Raclet, E. Badouel, A. Benveniste, B. Caillaud, R. Passerone, Why are modalities good for interface theories? in: Application of Concurrency to System Design, ACSD, IEEE Computer Society Press, 2009, pp. 119–127.
[43] J.-B. Raclet, Quotient de spécifications pour la réutilisation de composants, Ph.D. Thesis, Université de Rennes I, december 2007 (in French).
[44] J.-B. Raclet, Residual for component specifications, in: Formal Aspects of Component Software, FACS, in: Electr. Notes Theor. Comput. Sci., vol. 215, 2008, pp. 93–110.
[45] P. Bhaduri, Synthesis of interface automata, in: Automated Technology for Cerification and Analysis, ATVA, in: Lecture Notes in Computer Science, vol. 3707, 2005, pp. 338–353.
[46] A. David, K.G. Larsen, A. Legay, U. Nyman, A. Wąsowski, Timed i/o automata: a complete specification theory for real-time systems, in: Hybrid Systems: Computation and Control, HSCC, ACM, 2010, pp. 91–100.

[47] N. Bertrand, S. Pinchinat, J.-B. Raclet, Refinement and consistency of timed modal specifications, in: Language and Automata Theory and Applications, LATA, in: Lecture Notes in Computer Science, vol. 5457, Springer, Tarragona, Spain, 2009, pp. 152–163.
[48] N. Bertrand, A. Legay, S. Pinchinat, J.-B. Raclet, A compositional approach on modal specifications for timed systems, in: International Conference on Formal Engineering Methods, ICFEM, in: Lecture Notes in Computer Science, vol. 5885, Springer, 2009, pp. 679–697.
[49] M.Z. Kwiatkowska, G. Norman, J. Sproston, F. Wang, Symbolic model checking for probabilistic timed automata, Inform. and Comput. 205 (7) (2007) 1027–1077.
[50] B. Delahaye, B. Caillaud, A. Legay, Probabilistic contracts: A compositional reasoning methodology for the design of stochastic systems, in: 10th International Conference on Application of Concurrency to System Design, ACSD 2010, Braga, Portugal, 21–25 June 2010, IEEE Computer Society, 2010, pp. 223–232.
[51] L. de Alfaro, T.A. Henzinger, R. Jhala, Compositional methods for probabilistic systems, in: Concurrency Theory, CONCUR, in: Lecture Notes in Computer Science, vol. 2154, Springer, 2001, pp. 351–365.