Mixed Nondeterministic-Probabilistic Automata

Albert Benveniste, Jean-Baptiste Raclet

Noname manuscript No.
(will be inserted by the editor)

Mixed Nondeterministic-Probabilistic Automata

Blending graphical probabilistic models with nondeterminism

Albert Benveniste · Jean-Baptiste Raclet

September 11, 2023

Abstract Graphical models in probability and statistics are a core concept in the area of probabilistic reasoning and probabilistic programming—graphical models include Bayesian networks and factor graphs. For modeling and formal verification of probabilistic systems, probabilistic automata were introduced.

This paper proposes a coherent suite of models consisting of *Mixed Systems, Mixed Bayesian Networks,* and *Mixed Automata,* which extend factor graphs, Bayesian networks, and probabilistic automata with the handling of nondeterminism. Each of these models comes with a parallel composition, and we establish clear relations between these three models. Also, we provide a detailed comparison between Mixed Automata and Probabilistic Automata.

Keywords probabilistic automata; semantics; nondeterminism and probability; factor graphs; Bayesian networks; probabilistic programming

This work was supported by the project ReaLiSe, Émergence Ville de Paris 2021 - 2025.

Albert Benveniste
Inria Rennes Bretagne Atlantique, 35042 Rennes Cedex, France
Tel.: +33299847235
E-mail: albert.benveniste@inria.fr

Jean-Baptiste Raclet
IRIT, Université de Toulouse, CNRS, Toulouse INP, UT3, Toulouse, France
E-mail: jean-baptiste.raclet@irit.fr

Contents

1 Introduction

1.1 *Context*

Bayesian *graphical modeling* and inference [26] have expanded since the 1980s, with applications in numerous areas. Graphical models were introduced in probability and statistics to allow for a modular description of models [48]. Graphical models divide into two subfamilies: (directed) Bayesian Networks proposed initially by Judea Pearl [50] to support causal reasoning and (nondirected) Factor Graphs [40,43,48]. Probabilistic graphical modeling gave birth to an important sub-community of probabilistic programming [44,52,48].

Factor Graphs allow for the modular specification of unnormalized probabilities through a nondirected bipartite graph relating variables and factors via edges. To each factor is attached a local probability on the set of its neighboring variables. Taking the product of these local probabilities yields an unnormalized probability distribution. Logarithms of probabilities are often considered instead and added under the name of potential [40].

Bayesian Networks are causal graphical probabilistic models. The specification of causality comes extra to the specification of the underlying probability distribution in the form of directed branches of the graph. Judea Pearl [51] pointed out that causality is extra information relating random variables, not inferrable from their joint probability distribution.

Message passing algorithms are a crucial tool for Factor Graphs, allowing to map a subclass of them to Bayesian networks; see [43] and Section 3.6 of [48].

Through the union of underlying graphs and the compositional nature of probabilities specified by graphical probabilistic models, both Bayesian Networks and Factor Graphs frameworks are naturally equipped with some *parallel composition.* These features explain why graphical models are considered an intermediate format targeted by some probabilistic programming tools, see, e.g., [44,52], and [48], chapter 3.

Probabilistic Programming [44,52,19,49,39,48,9] supports specifying statistical models with modularity and libraries for performing inference. Some probabilistic languages generate *likelihood functions* [52,44,19] to be used by inference algorithms, whereas others generate *sampling* procedures [32,33]. Recently, G. Baudart et al. [9] proposed *reactive probabilistic programming* of dynamical systems as a conservative extension of synchronous languages [12] by enhancing the Hybrid Systems modeling language Zelus [11] with probabilities. Therefore, the objectives of Probabilistic Programming can be categorized as follows:

1. *Modeling paradigm.* Blending probability and nondeterminism, composing, and comparing (equivalence) are the main issues.
2. *Model for proof systems.* Calculi and their decidability and complexity are central issues in this objective.

3. *Support for statistical inference, decision, and learning.* Key pillars are all limit theorems of probability and statistics (law of large numbers, central limit theorem, large deviations). These theorems rely on the underlying probabilistic model's stationarity (or time invariance). For models with no dynamics, independent identically distributed (i.i.d.) sets of data can be sampled from the model. For models with dynamics (e.g., Markov chains), runs can be observed and used to infer model characteristics. One central difficulty in this objective is blending nondeterminism with probabilities, as it generally breaks the stationarity of the underlying statistical model. For example, suppose two different statistical models are combined with a nondeterministic choice (or an if-then-else statement with a nondeterministic guard). In that case, the stationarity of the overall model no longer holds. The solution is to recover stationary models by separating the two alternatives and not mixing them. See, e.g., [37] for related developments.

Concerning the context of probabilistic programming, our work focuses on objective 1 only, with no consideration of other objectives.

A critical issue in several contexts is *the combination of probabilistic and nondeterministic* behaviors:

- In *statistical decision* procedures, deciding whether the distribution of an observed sample belongs to subset $\mathcal{P}_1$ or $\mathcal{P}_2$ of probability distributions (where these subsets have empty intersection) exhibits nondeterminism in that the actual distribution is freely chosen within one of the two alternatives; this blending of probabilistic and nondeterministic behaviors is addressed in this case by using generalized likelihood ratio (GLR) tests [42].
- The formal verification community proposed the model of *Probabilistic Automata* (PA) [56,45,58,55]. In one of the PA dialects, runs of such PA proceed as follows: given a pair $\{\text{state}, \text{action}\}$, a nondeterministic choice among a finite set of probabilistic states is first performed, and then the corresponding probability distribution is sampled to derive the next state.
- Finally, mixing probabilistic behaviors and nondeterminism is central to probabilistic programming [46,20,47].

In this work, we propose a comprehensive suite of *mathematical models supporting the blending of probability and nondeterminism.* Motivated by engineering applications, we bear in mind modeling frameworks such as (the discrete time part of) Simulink or Modelica.

1.2 *Running example*

To illustrate our purpose, we begin with a running example whose style is motivated by the above considerations.

Throughout this paper, all variables possess finite or denumerable types—this restriction is motivated by technical reasons explained later. Hence, types

will not be declared when presenting examples. In the "if-then-else" statements, it is understood that the control variable is Boolean. Consider the following discrete time dynamical system (universal quantifier $\forall n$ is implicit):

$$S_1 : \begin{cases} x_0 = c_x \\ \texttt{observe}\ u_n \\ x_n = \varphi(u_n, x_{n-1}) \\ y_n = \text{if } f_n \text{ then } \psi(x_n, v_n) \text{ else } x_n \end{cases} \tag{1}$$

Model (1) involves *signals,* i.e., sequences, indexed by the natural integer n, of variables having the same type: for instance, signal x_n denotes the sequence $\{x_k \mid k{\in}\mathbb{N}\}$. The special statement `observe` u_n, involving the keyword `observe`, specifies that signal u_n is observed, meaning that that some oracle gives its sequence of values.

The intuition of model (1) is the following: f_n is a boolean signal indicating the occurrence of a failure and v_n is a noise, i.e., some disturbance. When a failure occurs, signal x_n gets corrupted by noise v_n, which is captured by the (unspecified) function ψ; otherwise, $y_n = x_n$. Since model (1) involves the delayed signal x_{n-1}, an initial condition for this signal is specified by $x_0 = c_x$, where c_x is some constant of same type as signal x_n. Model (1) is a dynamical system as usual, with inputs u, f, and v, state x, and output y. Unspecified functions φ and ψ are assumed total.

Our special syntax, however, suggests that we are interested in a different interpretation, by which model (1) specifies what is observed/unobserved: u_n is observed at every instant, whereas other signals are unobserved (the default case). From this perspective, signals f, v, x, and y are unknown and otherwise subject to (1). Thus, model (1) involves *nondeterminism.*

Next, consider the following stochastic model for noise v_n:

$$S_2 : v_n \sim \mu \tag{2}$$

where $v_n \sim \mu$ means that variable v_n has distribution μ at each instant n. As an essential convention of our modeling framework, statement $v_n \sim \mu$, taken in isolation, also means that the random sequence v_n is *independent, identically distributed (i.i.d.).* No signal is observed in this model (capturing that we are considering an unobserved disturbance).

Having the two models S_1 and S_2, we like to *compose* them, thus considering S_1 `and` S_2, defined as the conjunction of the two systems of equations (1,2). S_1 `and` S_2 combines stochastic behavior with nondeterminism (since failure signal f_n is still unknown and unobserved). As a consequence of this composition, the nature of signal y_n may or may not involve randomness due to the if-then-else statement occurring in S_1.

Consider next the following S_3 model specifying the behavior of the failure signal f:

$$S_3 : \begin{cases} f_0 = \texttt{F} \\ f_n = (rf_n \text{ and not } bk_n) \text{ or } f_{n-1} \\ rf_n \sim \beta =_{\text{def}} \text{Bernoulli}(10^{-6}) \end{cases} \tag{3}$$

In this model, "root failure" signal rf is modeled as a Bernoulli sequence, i.e., $P(rf = \mathtt{T}) = \beta(\mathtt{T}) = 10^{-6}$; boolean signal bk indicates that a "backup sensor" is provided. Thus, a failure is raised ($f = \mathtt{T}$) if a root failure occurs and no backup sensor is provided, and it remains subsequently raised. In S_3, no signal is observed, thus bk is nondeterministic. Model (3) is mixed probabilistic/nondeterministic. If bk was specified as being observed, this model would become probabilistic in that, once the value of random signal rf_n is known, the actual value of f_n is determined.

The next step is to further compose S_1 `and` S_2 with S_3. By convention of the parallel composition, as a consequence of composing the two statements "$v_n \sim \mu$" and "$rf_n \sim \beta$", *the two random sequences* v_n *and* rf_n *are a priori mutually independent.*

As a safety issue, we could be interested in evaluating the risk of missing an alarm raised by having signal y exceeding some threshold: an alarm is raised when $y_n > y_{\max}$. This alarm triggers some reconfiguration which is not shown here. This reconfiguration action was designed to act under the hypothesis that the system is fault-free, i.e., $y_n = x_n$ always holds. Consider the following question: what is the "risk" that an alarm is missed when it should have occurred, due to a fault? More precisely,

$$\text{what is the risk that "}x_n > y_{\max} \text{ and } y_n \leq y_{\max}\text{" occurs?} \tag{4}$$

So far, we did not define what we mean by "risk". A probability cannot measure it, since S_1 `and` S_2 `and` S_3 mixes probability with nondeterminism. By "risk", we mean a pessimistic evaluation of this probability, with nondeterminism acting as an adversary.

Suppose, next, that we want to specify that signal y is observed in the system S_1 `and` S_2 `and` S_3. To this end, we consider the system

$$S_4 : \texttt{observe}\ y_n \tag{5}$$

where no dynamics is otherwise specified. Parallel composition S_1 `and` S_2 `and` S_3 `and` S_4 expands as the following model (equation labels sitting on the right column are for subsequent use):

$$\begin{cases} x_0 = c_x\,,\ f_0 = \mathtt{F} & (e_0) \\ \texttt{observe}\ u_n & (e_1) \\ \texttt{observe}\ y_n & (e_2) \\ x_n = \varphi(u_n, x_{n-1}) & (e_3) \\ y_n = \text{if } f_n \text{ then } \psi(x_n, v_n) \text{ else } x_n & (e_4) \\ f_n = (rf_n \text{ and not } bk_n) \text{ or } f_{n-1} & (e_5) \end{cases} \tag{6}$$

$$\begin{cases} rf_n \sim \beta & (p_1) \\ v_n \sim \mu & (p_2) \\ (rf_n \text{ and } v_n \text{ are mutually independent i.i.d. signals}) \end{cases} \tag{7}$$

The intended semantics of model (6,7) is as follows: (7) specifies the *prior distribution* of the pair (v, rf) of random signals, where, by convention, the

two signals are considered independent. (6) defines a constraint on the set of variables involved in the system. The `observe` constraint on the pair u, y states that its joint trajectory is given by an oracle (the sensors). Consequently, the pair (v, rf) of random signals is now equipped with the *posterior distribution* resulting from constraint (6) being enforced.

Use of the running example: the above example will be referred to in the sequel by regarding it in two different ways:

> *Static setting:* in this setting, we focus on the transition relation: the initial conditions are discarded, and the values of delayed signals are considered as fixed; for example, in equation $x_n = \varphi(u_n, x_{n-1})$, the value of x_{n-1} is given and the equation specifies a constraint involving the two variables x_n and u_n. (8)
>
> *Dynamic setting:* in this setting, time index n ranges over $\mathbb{N}$ and the models (1)–(7) are considered as specifying dynamical systems. (9)

If we regard systems $S_1, \ldots, S_4$ as boxes with wires (the involved signals), this modeling approach naturally leads to graphical models similar to Factor Graphs. Indeed, this way of specifying mixed probabilistic/nondeterministic systems is fully modular: component models can be freely assembled to yield system models. Primitive statements are: 1) declarations of prior distributions; 2) declarations of constraints on signals through equations relating them, implicitly resulting in the definition of a posterior distribution; and 3) a parallel composition in which composing prior distributions considers them independent, and systems of equations are composed as usual. Closest to this approach are [9] (born from synchronous programming [12]) or [35] (born from concurrent constraint programming).

1.3 *Contribution*

As a semantic domain for the above modeling approach, we propose a comprehensive suite of mathematical frameworks blending probability and nondeterminism, consisting of: *Mixed Probabilistic Systems, Mixed Bayesian Networks,* and *Mixed Automata*—the term "mixed" refers to the blending with nondeterminism. We focus on semantics issues, such as: What is the probabilistic model specified? Given seemingly different system specifications, are they equivalent, or do they differ? Can one define a parallel composition of models?

Initially proposed in [13], Mixed Probabilistic Systems, termed *Mixed Systems* in the sequel, are pairs consisting of a private probability space and a visible state space, related through a relation. This pair specifies a posterior distribution, namely the conditional distribution, given that the relation between states and random outcomes is satisfied. Visible states are exposed for possible interaction with other Mixed Systems. Mixed Systems are equipped with a parallel composition. We show that Mixed Systems naturally inherit a notion of graphical structure, which subsumes Factor Graphs. Mixed Systems

offer the counterpart of angelic/demonic nondeterminism [20] and hard/soft conditioning [59,62], which are essential notions in probabilistic programming. In basic probability theory, kernels (also called transition probabilities) are functions, mapping each point of a set to a probability. Kernels are used to reason on conditional probabilities. Networks of kernels are called Bayesian Networks, extensively used in Bayesian reasoning. We propose *Mixed Kernels* and *Mixed Bayesian Networks* as mild extensions of kernels and Bayesian Networks to encompass nondeterminism. We study how Mixed Systems and Mixed Bayesian Networks relate to each other. So far, the above frameworks do not support recursion, even in the restricted form combining time and dynamics. Probabilistic Automata [58] derive from automata by considering transitions mapping states to probabilistic states, from which the next state is sampled. Replacing probabilities by Mixed Systems in a similar construction, we propose *Mixed Automata* as an extension of Automata supporting probability and nondeterminism.

The paper is organized as follows. Mixed Systems are introduced and further studied in Section 2. Mixed Automata are introduced and studied in Section 4, and then compared to Probabilistic Automata in Section 5. We use this apparatus in Section 6 to give the semantics of mini-language ReactiveBayes formalizing our running example. Related work is discussed more broadly in Section 7. Missing proofs are deferred to appendices. Focused bibliographical discussions are presented following each important notion. The reason is that the same mathematical notion occurs in different communities under different names. Hence, we felt it worthwhile to relate them.

2 Mixed Systems, parallel composition, and Factor Graphs

$\mathcal{X}$ shall denote an underlying set of *variables,* of finite domain. Elements of $\mathcal{X}$ are denoted by lower case letters $x, y, z \ldots$, and finite subsets of $\mathcal{X}$ are denoted by upper case letters X, Y, Z. We use set theoretic operations on sets of variables. The domain of x is denoted by Q_x and the domain of X is $Q_X =_{\text{def}} \prod_{x \in X} Q_x$, we call it the *state space*; the generic element of Q_X is called a *state* and is denoted by q_X or simply q.

The pair (Ω, π) shall denote a discrete probability space. The elements of Ω are denoted by the symbol ω. Thus, π is a countably additive function, from 2^Ω to $[0, 1]$, such that $\pi(\emptyset) = 0$ and $\pi(\Omega) = 1$. We simply write $\pi(\omega)$ instead of $\pi(\{\omega\})$. The *support* of π is the set $\mathbf{supp}(\pi) =_{\text{def}} \{\omega \mid \pi(\omega) > 0\}$. For a subset $W \subseteq \Omega$ such that $\pi(W) > 0$, the *conditional probability* $\pi(V|W) =_{\text{def}} \frac{\pi(V \cap W)}{\pi(W)}$ is well defined.

Finally, we will consider *relations* (or *constraints*) $C \subseteq \Omega \times Q$. Relations are composed by intersection.

Disclaimer: in this paper, we consider only discrete probability spaces. This restriction is technically important since it allows for a straightforward definition of conditional probabilities, and the notion of support of a probability

is easily defined. For the general case, the notion of conditional expectation is always defined [24], whereas conditional distributions require additional topological assumptions for their existence, and so does the notion of support. We decided not to cover those extensions to keep our work more straightforward.

2.1 *Mixed Systems, definition and semantics*

In this section we introduce Mixed Systems and show that they extend and subsume in a unified framework: nondeterminism, probability spaces, and factor graphs. This section is inspired in part by [13].

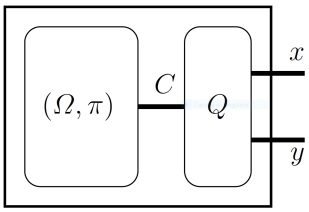


Fig. 1 Intuitive picturing of a Mixed System having two variables x and y.

The intuition is illustrated in Figure 1, which will guide us through the different notions attached to Mixed Systems. A Mixed System will be a pair consisting of a probability space (Ω, π) and a state space Q collecting the configurations of a set of state variables (here: x and y), related by a relation C. The probability space is "private" in that it is not directly exposed to any interaction with the environment. Interactions with the environment only occur through the state variables, thus seen as "visible". This distinction between private/visible is shown in Figure 1 by the outgoing pins x, y, which contrast with the absence of outgoing pin for the probabilistic box. The diagram of Figure 1 specifies

$$\begin{array}{l}\text{a probability distribution of}\\ \text{nondeterministic choices over a set of states.}\end{array} \tag{10}$$

This intuition is formalized next.

Definition 1 (Mixed System, consistency) A *Mixed System* (or *system* for short) is a tuple $S = (\Omega, \pi, X, C)$, where: (Ω, π) is a probability space; X is a finite set of variables with domain $Q = \prod_{x \in X} Q_x$; and $C \subseteq \Omega \times Q$ is a relation. In the sequel, we write

$$\omega C q$$

to mean $(\omega, q) \in C$. S is called *consistent* if

$$\pi(\Omega^{\mathsf{c}}) > 0\,,$$

where $\Omega^{\mathsf{c}} =_{\text{def}} \{\omega \in \Omega \mid \exists q : \omega C q\}$, called the *consistent subset,* is the projection of C over Ω.

Example 1 (Specializing to pure nondeterministic systems) A pure nondeterministic system is specified as a subset $\widehat{C} \subseteq Q$ of the state space. To reformulate it as a Mixed System, take (Ω, π) trivial, i.e., $\Omega = \{\omega\}$, a singleton, equipped with the trivial probability such that $\pi(\omega) = 1$, and define $\omega C q$ iff $q \in \widehat{C}$. Consider the running example of Section 1.2 in its static setting (8). Then, system (1), or any single equation involved in this system, are examples of constraint $\widehat{C}$ giving rise to such pure nondeterministic systems.

Example 2 (Specializing in pure probabilistic systems) A pure probabilistic system is specified as a pair (Ω, π). To reformulate it as a Mixed System, take $Q = \Omega$, and let C be the diagonal of $\Omega \times Q$; finally, let x be the variable with domain Q. Referring to the running example of Section 1.2 in its static setting (8), system (2) and (7) are examples giving rise to such a pure probabilistic system.

Example 3 (Nondeterministic choice between probabilities) Referring to (10), one more traditional way of mixing probabilities and nondeterminism is by considering the nondeterministic choice between probabilities, i.e., for example, the nondeterministic choice between two different distributions π_1 and π_2 defined over the same finite set W. Sampling this model consists in 1) nondeterministically selecting π_1 or π_2, and 2) drawing at random $w \in W$ with the selected probability. We capture this by the following Mixed System: $\Omega = W \times W$; $\pi = \pi_1 \times \pi_2$; $Q = \{1, 2\} \times W$, and

$$C = \big\{ \big((w_1, w_2), (1, w_1)\big)\,,\ \big((w_1, w_2), (2, w_2)\big) \mid (w_1, w_2) \in \Omega \big\}$$

This method for mapping the nondeterministic choice between probabilities, to Mixed Systems, extends to any finite number of probabilities. □

Example 4 (inconsistent system) Let $C \subseteq \Omega \times Q$ and $D \subseteq \Omega$ be such that $(D \times Q) \cap C = \emptyset$, and let π be the uniform probability over D. Then, the system $S = (\Omega, \pi, X, C)$ is inconsistent: sampling π will never return an ω such that $\exists q. \omega C q$ is satisfiable, i.e., the system is self-contradicting. Indeed, the same holds for any π such that $(\mathbf{supp}(\pi) \times Q) \cap C = \emptyset$. □

We are interested in understanding how Mixed Systems are "executed" and how state properties—which are not random—can still get some probabilistic evaluation. This is answered next by the *sampling semantics* and *probabilistic semantics,* which will be defined only for consistent systems.

Definition 2 (Sampling semantics) If S is consistent, its *sampling* is well defined as follows. Considering the *posterior probability* π^{c}, defined by

$$\forall A \subseteq \Omega : \pi^{\mathsf{c}}(A) \ =_{\text{def}}\ \pi(A \mid \Omega^{\mathsf{c}}) = \tfrac{\pi(A \cap \Omega^{\mathsf{c}})}{\pi(\Omega^{\mathsf{c}})}\,; \tag{11}$$

1. sample $\omega \in \Omega$ according to posterior probability π^{c};
2. nondeterministically select $q \in Q$ such that $\omega C q$.

This two-step procedure is denoted by $S \rightsquigarrow q$. □

Unlike for example 4, step 2 in the definition of the sampling is, by construction, always satisfiable.

Example 5 (sampling) We illustrate sampling on Example 3. The system probability is $\pi = \pi_1 \times \pi_2$ and $\pi^{\mathsf{c}} = \pi$ for this example. Hence, sampling this system proceeds as follows:

1. Sample independently π_1 and π_2, which yields w_1 and w_2, respectively.
2. Select nondeterministically state 1 or 2; if 1 is selected, then produce the pair $(1, w_1)$ and similarly if 2 is selected.

One of the two independent samplings (of π_1 and π_2) is useless in our sampling scheme, and could be discarded. Thus, sampling this example could equivalently proceed by 1) nondeterministically selecting state 1 or 2, and then drawing w from π_1 or π_2, accordingly.

Definition 3 (Probabilistic semantics) If S is consistent, its *probabilistic semantics* is defined as the pair $\overline{\pi}, \underline{\pi} : 2^Q \to [0,1]$, where, for any state property $A \subseteq Q$:

$$\overline{\pi}(A) =_{\text{def}} \pi^{\mathsf{c}}(\Omega_{\exists A}) \text{ where } \Omega_{\exists A} =_{\text{def}} \{\omega \in \Omega \mid \exists q \in A : \omega C q\} \tag{12}$$

$$\underline{\pi}(A) =_{\text{def}} \pi^{\mathsf{c}}(\Omega_{\forall A}) \text{ where } \Omega_{\forall A} =_{\text{def}} \{\omega \in \Omega \mid \forall q \in A : \omega C q\} \tag{13}$$

$\overline{\pi}$ defined by formula (12) is not a probability on Q, but only an *outer probability*[1], i.e., a function $\overline{\pi} : 2^Q \to [0,1]$ such that $\overline{\pi}(\emptyset) = 0$, $\overline{\pi}(Q) = 1$, and $\overline{\pi}$ is sub-additive, meaning that it satisfies

$$\forall A, (A_n)_{n\in\mathbb{N}} \text{ subsets of } Q : A \subseteq \bigcup_{n\in\mathbb{N}} A_n \implies \overline{\pi}(A) \leq \sum_{n\in\mathbb{N}} \overline{\pi}(A_n)\,.$$

Similarly, $\underline{\pi}$ is an *inner probability*, i.e., it is super-additive. Note that,

$$\text{if } A = \{q\} \text{ is a singleton, then } \overline{\pi}(A) = \underline{\pi}(A). \tag{14}$$

Example 6 (probabilistic semantics) We now illustrate outer probability $\overline{\pi}$ on Example 3. Let $W' \subseteq W$ and consider the state property $A_1 = \{1\} \times W' \subseteq Q$ of the state space (subset $A_2 = \{2\} \times W'$ is handled similarly). Recall that $\pi^{\mathsf{c}} = \pi_1 \times \pi_2$. Then, $\Omega_{\exists A_1} = \{(w_1, w_2) \mid w_1 \in W'\}$, hence, $\overline{\pi}(A_1) = (\pi_1 \times \pi_2)(W' \times W) = \pi_1(W')$.
For $A = A_1 \cup A_2 = \{1,2\} \times W'$, we get $\Omega_{\exists A} = \{(w_1, w_2) \mid w_1 \in W' \text{ or } w_2 \in W'\}$, hence, $\overline{\pi}(A) = (\pi_1 \times \pi_2)\big((W' \times W) \cup (W \times W')\big)$, which is $\leq \overline{\pi}(A_1) + \overline{\pi}(A_2)$. For the inner probability, $\Omega_{\forall A_1} = \emptyset$, hence $\underline{\pi}(A) = 0$. □

[1] sometimes called also *exterior* or *upper* probability.

For inference purposes, the following *generalized likelihood* $\ell : 2^{Q} \to [0,1]$ is also of interest:

$$\ell(A) \;=_{\text{def}}\; \max_{\omega \in \Omega_{\exists A}} \; \pi^{\mathsf{c}}(\omega)\,. \tag{15}$$

Note that (15) resembles (12) if we rewrite the latter as $\overline{\pi}(A) = \sum_{\omega \in \Omega_{\exists A}} \pi^{\mathsf{c}}(\omega)$.

Example 7 (probabilistic semantics, running example) Consider the model introduced in Section 1.2 denoted S_1 and S_2 and S_3 , interpreted in its static setting (8). Pick an instant n and let $S \;=_{\text{def}}\; S(n, x_{n-1}, f_{n-1})$ be the Mixed System defined by (1,2,3) for instant n and given values for x_{n-1}, f_{n-1}. With reference to (4), we wish to evaluate the probability that $x_n > y_{\max}$ and $y_n \leq y_{\max}$ occurs under adversarial nondeterminism. We denote by Q_v the domain of v and by $\mathbb{B}_{rf}$ the Boolean domain for variable rf. The underlying probability space of S is (Ω, π), where $\Omega = Q_v \times \mathbb{B}_{rf}$ and $\pi = \mu \times \beta$. Domain Q for the variables of S is

$$Q \;\; = \;\; Q_x \times Q_y \times Q_u \times Q_v \times \mathbb{B}_{rf} \times \mathbb{B}_f \times \mathbb{B}_{bk}\,,$$

and relation C is defined by collecting the nonprobabilistic equations of S while discarding the initial conditions, i.e.,

$$C \text{ is defined by } \begin{cases} \texttt{observe}\ u_n \\ x_n = \varphi(u_n, x_{n-1}) \\ y_n = \text{if } f_n \text{ then } \psi(x_n, v_n) \text{ else } x_n \\ f_n = (rf_n \text{ and not } bk_n) \text{ or } f_{n-1} \end{cases} \tag{16}$$

For any valuation of the pair (rf_n, v_n), a solution exists for state variables in (16). Hence, $\Omega^{\mathsf{c}} = \Omega$, which implies $\pi^{\mathsf{c}} = \pi = \mu \times \beta$.

Let $C(u_n)$ denote the relation C in which the value of u_n is given (u is observed). Then

$$\begin{array}{l} \overline{\pi}(x_n{>}y_{\max} \text{ and } y_n{\leq}y_{\max}) = \pi^{\mathsf{c}}(W) = (\mu{\times}\beta)(W)\,, \quad \text{where} \\ W = \left\{ (v_n, rf_n) \;\middle|\; \exists x_n, y_n, f_n, bk_n : \begin{array}{l} x_n{>}y_{\max} \text{ and } y_n{\leq}y_{\max}\,, \text{ and} \\ (x_n, y_n, v_n, rf_n, f_n, bk_n) \in C(u_n) \end{array} \right\} \end{array} \tag{17}$$

Inspecting (16) shows that condition $x_n{>}y_{\max}$ and $y_n{\leq}y_{\max}$ requires $f_n = \mathtt{T}$, since, if $f_n = \mathtt{F}$, then $y_n = x_n$. Hence, the condition defining W rewrites as

$$\begin{cases} x_n{>}y_{\max} \text{ and } \psi(x_n, v_n){\leq}y_{\max} \text{ and } f_n{=}\mathtt{T} \\ \text{and } (x_n, y_n, v_n, rf_n, f_n, bk_n) \in C(u_n) \end{cases} \tag{18}$$

First, if $\varphi(u_n, x_{n-1}) \leq y_{\max}$ holds, then $W = \emptyset$. We thus assume in the sequel $\varphi(u_n, x_{n-1}) > y_{\max}$. Thus we need to evaluate with respect to $\overline{\pi}$ the predicate

$$Z \;\; =_{\text{def}} \;\; \psi(x_n, v_n){\leq}y_{\max} \text{ and } f_n{=}\mathtt{T} \text{ and } (x_n, y_n, v_n, rf_n, f_n, bk_n) \in C(u_n)\,.$$

We distinguish the following two cases, according to the value of f_{n-1}:

1. $f_{n-1}{=}\mathtt{T}$: then, $f_n{=}\mathtt{T}$ whatever the value of bk_n is, and, using (18):

$$\overline{\pi}(x_n{>}y_{\max} \text{ and } y_n{\leq}y_{\max}) = \mu\big\{v \mid \psi(\varphi(u_n, x_{n-1}), v) \leq y_{\max}\big\}\,.$$

2. f_{n-1}=F: then, f_n=T if and only if rf_n=T and bk_n=F. For the nondeterministic choice of the state, we thus chose bk_n=F. By definition of the outer probability (12), we finally get, using (18):

$$\overline{\pi}(x_n{>}y_{\max} \text{ and } y_n{\leq}y_{\max}) = \beta(rf{=}\mathtt{T}) \times \mu\big\{v \mid \psi(\varphi(u_n, x_{n-1}), v) \leq y_{\max}\big\}\,.$$

This corresponds to the probabilistic evaluation of the predicate "$x_n{>}y_{\max}$ and $y_n{\leq}y_{\max}$" following [20], if the nondeterministic alternative bk_n=F/T is interpreted as demonic. See Discussion 1 for further comparison with the literature. □

Equivalence: We move to equivalence. To this end, we introduce the following operation of compression, on top of which equivalence is defined:

Definition 4 (compression) For $S = (\Omega, \pi, X, C)$ a Mixed System, we define the following equivalence relation on Ω, i.e., $\sim \subseteq \Omega{\times}\Omega$ is such that:

$$(\omega, \omega') \in \sim \text{ if and only if: } \forall q \in Q : \omega C q \Leftrightarrow \omega' C q\,. \tag{19}$$

As usual, we write $\omega \sim \omega'$ to mean $(\omega, \omega') \in \sim$. The *compression* of S, denoted by $\widetilde{S} = (\widetilde{\Omega}, \widetilde{\pi}, X, \widetilde{C})$, is then defined as follows:

- $\widetilde{\Omega}$ is the quotient $\Omega/{\sim}$, which elements are written $\widetilde{\omega}$;
- $\widetilde{\omega}\widetilde{C}q$ iff $\omega C q$ for $\omega \in \widetilde{\omega}$; and
- $\widetilde{\pi}(\widetilde{\omega}) = \sum_{\omega\in\widetilde{\omega}} \pi(\omega)$.

Say that S is *compressed* if it coincides with its compression. □

Distinguishing ω and ω' is impossible if $\omega{\sim}\omega'$. Equivalence is defined on top of compression. For C and π as before, define

$$C_\pi \;=_{\text{def}}\; C \cap (\mathbf{supp}(\pi){\times}Q) = \{(\omega, q) \in C \mid \pi(\omega) > 0\}$$

Definition 5 (equivalence) Two compressed mixed systems S and S' are *equivalent* if they possess identical sets of variables $X{=}X'$, and there exists a bijective map $\varphi : C_\pi \mapsto C'_{\pi'}$ satisfying the following conditions for every pair $(\omega, q) \in \Omega \times Q$, where $(\omega', q') \;=_{\text{def}}\; \varphi(\omega, q)$:

$$\omega\, C_\pi\, q \Leftrightarrow \omega'\, C'_{\pi'}\, q' \;\;;\;\; \pi'(\omega') = \pi(\omega) \;\;;\;\; q' = q\,. \tag{20}$$

S and S' are *equivalent*, written $S{\equiv}S'$, if their compressions are equivalent.□

The following result expresses that mixed system equivalence preserves probabilistic semantics:

Lemma 1 *Any two equivalent mixed systems, $S_1 \equiv S_2$, possess identical probabilistic semantics: $\overline{\pi}_1 = \overline{\pi}_2$ and $\underline{\pi}_1 = \underline{\pi}_2$.*

Proof It is enough to prove the lemma in the following two cases: 1) S_1 and S_2 are both compressed, and 2): $S_2 = \widetilde{S}_1$. The result is immediate for case 1), so we focus on case 2). Let Q be the common domain of $X_1 = X_2$ and $A \subseteq Q$ be a state property. Then,

$$\begin{aligned}\overline{\pi}_1(A) &= \pi_1^{\mathsf{c}}\big(\{\omega_1 \mid \exists q \in A : \omega_1 C_1 q\}\big)\\ &= \pi_1^{\mathsf{c}}\big(\{\omega_1 \mid \omega_1 \in \widetilde{\omega}_1 \text{ and } \exists q \in A : \widetilde{\omega}_1 \widetilde{C}_1 q\}\big)\\ &= \widetilde{\pi}_1^{\mathsf{c}}\big(\{\widetilde{\omega}_1 \mid \exists q \in A : \widetilde{\omega}_1 \widetilde{C}_1 q\}\big) = \overline{\widetilde{\pi}}_1(A) = \overline{\pi}_2(A)\,.\end{aligned}$$

A similar proof holds for inner probabilities. □

Marginal: For (X, Y) a pair of random variables with joint distribution $P(x, y)$, the distribution of X is given by the *marginal* of P, namely: $P(x) =_{\text{def}} \sum_y P(x, y)$. We extend this notion to Mixed Systems, by viewing it as a hiding operation, see Figure 2. For $C \subseteq \Omega \times Q$ a relation where Q is the domain of

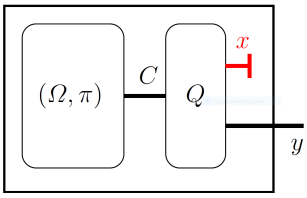


Fig. 2 The marginal on y for the Mixed System of Figure 1 is by hiding x (in red).

set X, $Y \subseteq X$ a subset of variables, and $Z = X - Y$, we denote by

$$\mathbf{Pr}_Y : 2^{\Omega \times Q} \to 2^{\Omega \times Q_Y} : \ \mathbf{Pr}_Y(C) =_{\text{def}} \{(\omega, q_Y) \mid \exists q_Z : \omega C(q_Y, q_Z)\}$$

the projection of C over Y.

Definition 6 (marginal) Let $S = (\Omega, \pi, X, C)$ be a Mixed System, and let $Y \subseteq X$ be a subset of variables. The *marginal* of S on Y, denoted by $\mathbf{Margin}_Y(S)$, is the Mixed System $\mathbf{Margin}_Y(S) =_{\text{def}} (\Omega, \pi, Y, \mathbf{Pr}_Y(C))$. □

Even if S was itself compressed, due to the projection of relation C, the Mixed System defining the marginal in Definition 6 may require a compression.

Example 8 (Link with the classical notion of marginal for probabilities) Let us apply Definition 6 to the purely probabilistic system of Example 2, namely $S_{\mathsf{proba}} = (Q, \pi, \{x, y\}, \mathit{diag})$, having two variables x, y, corresponding state space $Q = Q_x \times Q_y$, and $\Omega = Q$ with $C = \mathit{diag}$, the diagonal. This is the model of a pair (x, y) of visible variables with joint probability distribution $\pi(q_x, q_y)$, where q_x and q_y denote values for x and y, respectively. The projection of diag on y is

$$\mathbf{Pr}_y(\mathit{diag}) = \{(q_x, q_y, q'_y) \mid q_y = q'_y\}\,.$$

Thus, $(q_x, q_y) \sim (q'_x, q'_y)$ if and only if $q_y = q'_y$. Thus, when using the formula of Definition 6 to define $\mathbf{Margin}_y(S_{\mathsf{proba}})$, the private probability space (Q, π) must be compressed as $\widetilde{\pi}(q_y) = \sum_{q_x} \pi(q_x, q_y)$, showing that our notion of marginal boils down to the classical notion for probabilities in this case. □

2.2 *Parallel composition*

In this section, we study the modular construction of Mixed Systems.

In statistics, *Factor Graphs* allow for the modular specification of unnormalized probabilities based on a nondirected bipartite graph (V, F, E), where $V \cup F$ is the set of vertices and $E \subseteq V \times F$ is the set of edges; let V_f be the subset of $v \in V$ such that $(v, f) \in E$. V is a set of random variables, and, to each *factor* $f \in F$ is associated with an unnormalized probability $p_f(V_f)$ for the set V_f of random variables. This model defines the unnormalized probability distribution of V as the product $P(V) = \prod_{f \in F} p_f(V_f)$—logarithms of probabilities are often considered instead and added under the name of potential [40]. This naturally leads to a notion of parallel composition: for $(V_i, F_i, E_i), i = 1, 2$ two factor graphs in which the factors and edges are private but variables can be shared, taking the union of the two graphs amounts to taking the product of the two unnormalized probabilities.

Similarly, Mixed Systems can be equipped with a parallel composition: common state variables are unified (thus causing synchronization constraints); conversely, probabilistic parts remain local and independent, conditionally to the satisfaction of synchronization constraints. This is illustrated in Figure 3.

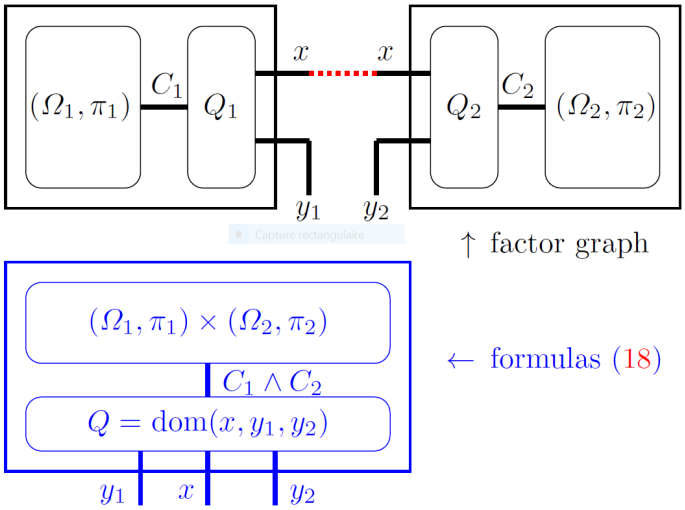


Fig. 3 Illustrating the parallel composition, for y_1, y_2 local variables and x shared. The factor graph, capturing the connection via identical wires, is depicted on the top in black; the definition using formulas (23) is shown in blue.

Formally, let I be a finite set, and, for each $i{\in}I$, let X_i be a finite set of variables with domain Q_i, and set $X = \bigcup_{i\in I} X_i$ with domain Q. Say that tuple $(q_i)_{i\in I}$ is *compatible*, written

$$\bowtie_{i\in I} q_i\,, \tag{21}$$

if $q_i(x) = q_j(x)$ for any pair (i,j) of indices and every shared variable $x \in X_i \cap X_j$. If $\bowtie_{i\in I} q_i$, their *join*

$$\sqcup_{i\in I} q_i \in Q \tag{22}$$

is defined by $\sqcup_{i\in I} q_i(x) = q_i(x)$ whenever $x \in X_i$.

Definition 7 (parallel composition and Factor Graph) The *parallel composition* $S_1 \parallel S_2$ of two mixed systems S_1 and S_2 is the Mixed System S such that:

$$\begin{array}{l} X = X_1 \cup X_2\,,\ \Omega = \Omega_1 \times \Omega_2\,,\ \pi = \pi_1 \times \pi_2 \ \text{ (cartesian product), and} \\ C = \big\{\big((\omega_1,\omega_2), q_1 \sqcup q_2\big) \mid q_1 \bowtie q_2 \text{ and } \omega_1 C_1 q_1 \text{ and } \omega_2 C_2 q_2\big\}\,, \end{array} \tag{23}$$

We attach to parallel composition $\mathcal{S} = \|_{i\in I} S_i$ its *Factor Graph* $\mathcal{G}_{\mathcal{S}}$, which is a *nondirected* bipartite graph whose vertices connect systems and variables:

$$\big\{S_i \mid i \in I\big\} \cup \big\{x \mid x \in \bigcup_{i\in I} X_i\big\},$$

and $\mathcal{G}_{\mathcal{S}}$ has edges (S_i, x), for $i \in I$ and $x \in X_i$, also denoted by $S_i - x$. □

Example 9 (parallel composition) Consider the running example of Section 1.2 in its static setting (8). Focus first on its pure nondeterministic part, collected in Model (6) with its equations (e_1)–(e_5). We can first consider (6) as a whole, and, by following the method of Example 1, encode it as a pure nondeterministic Mixed System denoted by $S_{(6)}$. Alternatively, we can consider the pure nondeterministic Mixed Systems $S_{(e_1)}, \ldots, S_{(e_5)}$, respectively encoding each of the equations $(e_1), \ldots, (e_5)$. By definition of our parallel composition, we have $S_{(6)} \equiv S_{(e_1)} \parallel \ldots \parallel S_{(e_5)}$, reflecting that pure nondeterministic systems compose as their associated constraints do. The reason for probabilistic parts playing no role in the parallel composition of pure nondeterministic systems, is that each $S_i =_{\text{def}} S_{(e_i)}, i{=}1,\ldots,5$, has its probabilistic part being a trivial singleton probability s ce of the form $(\{\omega_i\}, 1)$, and the Cartesian product $(\{\omega_1\},1){\times}\ldots{\times}(\{\omega_5\},1)$ is isomorphic to $(\{\omega\},1)$ for any ω.

Similarly, by following the method of Example 2 for the purely probabilistic part (7) of the running example, we have, using the same notational conventions, $S_{(7)} \equiv S_{(p_1)} \parallel S_{(p_2)}$, where the parallel composition corresponds to the Cartesian product $\mu{\times}\beta$ of the two constitutive probability distributions.

The parallel composition $S =_{\text{def}} S_{(6)} \parallel S_{(7)}$, which composes the nondeterministic and probabilistic parts, is more interesting. As discussed before, if S is the tuple (Ω, π, X, C), then (Ω, π) is such that $\pi = \mu{\times}\beta$ (the trivial probability distributions associated with the pure nondeterministic part $S_{(6)}$ play no role). In turn, $S_{(6)}$ contributes to the definition of the consistent subset

Ω^{c}, which is the subset of values for the pair (v_n, rf_n) for which there exists a solution to the system (e_1)–(e_5) of equations. The (sampling or probabilistic) semantics of S uses the posterior probability distribution $\pi^{\mathsf{c}} =_{\text{def}} \pi(\cdot|\Omega^{\mathsf{c}})$. Let us analyse how this posterior distribution relates to the prior one $\mu\times\beta$. Recall the system (e_1)–(e_5) for convenience:

$$\begin{cases} \texttt{observe}\ u_n & (e_1) \\ \texttt{observe}\ y_n & (e_2) \\ x_n = \varphi(u_n, x_{n-1}) & (e_3) \\ y_n = \text{if } f_n \text{ then } \psi(x_n, v_n) \text{ else } x_n & (e_4) \\ f_n = (rf_n \text{ and not } bk_n) \text{ or } f_{n-1} & (e_5) \end{cases} \tag{24}$$

If functions φ and ψ are total, then the subsystem (e_1, e_3, e_4, e_5) (e_2 was discarded) specifies a function $(u_n, v_n, rf_n, bk_n) \mapsto (f_n, x_n, y_n)$, meaning that no constraint is set on the pair (v_n, rf_n) by the consideration of Ω^{c}. Including (e_2) : `observe` y_n, however, changes the situation. The value of y_n is then known, and the existence of a solution to (24) induces a constraint on the tuple (u_n, v_n, rf_n, bk_n), which projects to the consistency constraint Ω^{c} on the pair (v_n, rf_n). The result is that, due to the statement `observe` y_n in (24), the posterior distribution π^{c} differs from the prior distribution $\pi = \mu\times\beta$. □

In the sequel, ϵ shall denote a distinguished state. Let

$$\mathsf{nil} = (\{1\}, \delta_1, \emptyset, \{(1, \epsilon)\}) \tag{25}$$

be the *nil* system, with trivial probability space $(\Omega, \pi) = (\{1\}, \delta_1)$ and no visible variable; its state space is the singleton $Q_{\mathsf{nil}} = \{\epsilon\}$, and its relation is the singleton $C = \{(1, \epsilon)\}$. The nil system is neutral for parallel composition: $\mathsf{nil} \parallel S \equiv S$ holds, for every S.

Factor Graphs obey the following rule, where $\cup$ denotes the union of graphs:

$$\mathcal{G}_{S_1 \parallel S_2} = \mathcal{G}_{S_1} \cup \mathcal{G}_{S_2}\,. \tag{26}$$

The associativity and commutativity of this parallel composition are immediate, as it is directly inherited from the same properties satisfied by the Cartesian product of probability spaces and the conjunction of relations. Factor Graphs and the parallel composition of Mixed Systems help decompose large but sparse systems into a parallel composition of smaller, locally interacting subsystems.

Lemma 2 *$S_1 \equiv S_1'$ implies $S_1 \parallel S_2 \equiv S_1' \parallel S_2$, expressing that parallel composition preserves equivalence.*

See Appendix A.2 for the proof.

2.3 *Discussion and comparison with related work*

We now collect technical discussions on the topics discussed in this section and formulate comparisons with related work from different communities.

Discussion 1 (blending nondeterminism and probability) To capture the blending of nondeterminism and probability, outer probabilities are directly used in the Dempster-Shafer theory of evidence [25,26,57].[2] Outer probabilities do not support key limit theorems for use in statistics, such as the law of large numbers, the central limit theorem, and more. Hence, whereas the theory of evidence comes with reasoning capabilities, it does not directly support learning or estimation.

In formal methods for probabilistic systems (in the context of imperative programming), the blending of probability and nondeterminism was addressed by a number of authors, see, e.g., [46,8,47,39,49,20,63]. Nondeterministic choice between alternatives is considered in [47] and written $P \sqcap P'$, whereas probabilistic choice is specified as $P \,{}_a\!\oplus P'$ (P is selected with probability a and P' with probability $1-a$) or $P \,{}_a\!\oplus_b P'$ (P is selected with probability at least a and P' with probability at least b). The evaluation of formulas must specify how nondeterminism interplays with probabilities. A comprehensive approach was proposed in [20], where *demonic* and *angelic* nondeterminisms are considered as adversarial and beneficial, respectively. These notions mirror the outer and inner probabilities used in Dempster theory.

Through formulas (12,13) in Definition 1, the probabilistic semantics of Mixed Systems is defined as the associated outer and inner probabilities. Hence, Mixed Systems offer the calculus of the theory of evidence, and mirror the demonic and angelic types of nondeterminism. But, on the other hand, since classical probability spaces are first class citizens of the model of Mixed Systems, this model also preserves the apparatus needed for machine learning. In Appendix A.1, we develop a more detailed comparison of the semantics of Mixed Systems versus imperative probabilistic programming with demonic and angelic nondeterminism, following [20].

Finally, the generalized likelihood of formula (15) is the basis for inference, estimation, or machine learning when multiple hypotheses or nuisance parameters are considered [42]—we are not aware of any use of a mirror notion where "min" would be substituted for "max". □

Discussion 2 (Conditioning and its variations) Conditioning is generally not considered in probabilistic automata. It is, however, central in probabilistic programming; see, e.g., [49,21,62] for studies in which conditioning is the main subject. The `observe` primitive, pervasive in all tools, is used to specify posterior distributions given constraints (as we do in Definition 1). The literature on probabilistic programming distinguishes between *hard* (also called *deterministic*) and *soft* (also called *stochastic*) conditioning [59,62]. In the basics of probability theory, however, the only notion is that of *conditional*

[2] Outer and inner probabilities were called upper and lower in [25].

expectation [24], from which other notions are derived, e.g., conditional probability (corresponding to hard conditioning), transition probability, or stochastic kernel (corresponding to soft conditioning), and disintegration (or regular version of conditional expectation). These notions are all straightforward in our case since we restrict ourselves to discrete probability spaces (Ω, π). Indeed, for $V, W \subseteq \Omega$, the hard conditioning of V knowing W is given by $\pi(V|W) =_{\text{def}} \frac{\pi(V \cap W)}{\pi(W)}$, which is well defined if and only if $\pi(W) > 0$, see the introductory paragraph of Section 2. If $x : \Omega \to Q_x$ is a random variable, then the soft conditioning of V knowing x is the function $\pi(V|x) : Q'_x \to [0, 1]$, such that $q_x \mapsto \pi(V|x{=}q_x)$, where $Q'_x = \{q_x \in Q_x \mid \pi(x{=}q_x) > 0\}$. We will discuss this further when extending Bayesian networks to Mixed Systems, in Section 3. □

Discussion 3 (more on consistency) Inconsistency formalizes the intuition of self-contradiction, for Mixed Systems, see Example 4. The condition "$\pi(\Omega^{\mathsf{c}}) > 0$" in Definition 1 means that Ω^{c} has non-empty intersection with the support of π, defined as the set of ω's of positive probability: $\pi(\omega) > 0$. This simple definition for the support, which is only valid for discrete probabilities, allows us to propose a simple definition for the notion of consistency. When continuous probability spaces are considered (like the Gaussian), the above definition for the support no longer holds. The correct definition relies on topological properties. As a consequence, our elementary definition of consistency would no longer apply. This is next illustrated by using our running example developed in the introduction.

Example 10 (consistency for a non-discrete (Ω, π)) Consider model (6,7). Definition 1 defines consistency as the existence of a state q in relation through C with some ω belonging to the support of π, which is fairly simple. Suppose, for a while, that u_n, x_n, y_n, v_n possess real domain, $\mu(dv) = \chi(v)dv$, where dv denotes the Lebesgue measure, density χ is continuous and everywhere positive, and function $v \mapsto \psi(x, v)$ is bijective and bi-continuous for every fixed x (the condition of bi-continuity is technical and is intended to avoid discussing issues of measurability). Then, fixing the value of y_n, for a given pair (u_n, x_{n-1}), will fix the value of v_n if $f_n{=}\mathtt{T}$ in the equation defining y_n. Regarding again Example 7, the only difference is that, in (6), a parallel composition with the statement `observe` y_n was added. Thus, we now consider $C(u_n, y_n)$ where the values of u_n and y_n are fixed. It still makes sense to consider the two cases 1 ($f_{n-1} = \mathtt{T}$) and 2 ($f_{n-1} = \mathtt{F}$) of Example 7. Regarding the set W, we are interested in the case when $y_n \leq y_{\max}$ holds. In case 1, we get

$$W = \left\{ (v_n, rf_n) \;\middle|\; \exists x_n, y_n, f_n, bk_n : \begin{array}{l} x_n > y_{\max} \text{ , and} \\ (x_n, v_n, rf_n, f_n, bk_n) \in C(u_n, y_n) \end{array} \right\}$$
$$\subseteq \left\{ (v_n, rf_n) \mid \psi(\varphi(u_n, x_{n-1}), v_n) = y_n \right\}$$

implying $(\mu \times \beta)(W){=}0$. Thus, according to our Definition 1 where consistency is defined, we should deduce that the system is inconsistent in case 1, because fixing the value of u_n, x_{n-1}, and y_n fixes the value of v_n, which has

μ-probability zero. However, the assumption on μ implies that its support is $\mathbb{R}$, which does not fit with inconsistency. This illustrates that our pedestrian definition of consistency no longer works if real variables and distributions having densities with respect to Lebesgue measure are considered. Hints toward a correct definition are presented in Appendix D of [14]. □

Discussion 4 (equivalence) Floyd/Hoare/Dijkstra logic of pre- and post-conditions for imperative languages was extended to encompass probability and nondeterminism with pGCL (probabilistic Guarded Command Language) [41,21,46,39,49,47]. The semantics is defined as the probability of weakest preconditions under demonic nondeterminism. McIver-Morgan notions of refinement and equivalence follow from this semantics. This approach is also used to define the equivalence of probabilistic programs, see, e.g., Section 3.1 of [48].

As pointed out in Discussion 1, the above semantics parallels our consideration of outer/inner probabilities in Definition 3. Compared to McIver-Morgan notion of equivalence, the notion of equivalence we propose in Definition 5 is more basic and direct. It implies the equivalence of the evaluation of state properties using outer/inner probabilities. □

3 Mixed Bayesian Networks and causal reasoning

So far, Factor Graphs and related algorithms can capture joint distributions relating to different statistical data, but they cannot capture causality, as argued by Judea Pearl [51]. Judea Pearl states that causality requires extra structural information that must be added to the specification of probability distributions: directed graphs are used to this end.

Another issue is that of incremental sampling of a compound system: Whereas the sampling of a parallel composition is generally global (or using the sophisticated iterative methods used, e.g., in [19]), one could ask whether it could be performed incrementally.

In statistics based on graphical models, these questions are answered by considering, in addition to Factor Graphs, so-called Bayesian Networks [50]. Bayesian networks specify causality information through directed graphs. A *Bayesian Network* is a tuple (V, E, p), where: V is a set of random variables; (V, E) is an acyclic directed graph (for each $v \in V$, we let $\text{PA}(v)$ denote its parents); $p(v|\text{PA}(v))$ specifies, for each valuation of the parents $\text{PA}(v)$, a conditional distribution for the variable v. The semantics of a Bayesian Network is that the joint distribution of V factorizes as the product $P(V) = \prod_{v \in V} p(v|\text{PA}(v))$. Bayesian Networks are thus causal graphical probabilistic models and the specification of causality comes extra to the specification of the underlying probability distribution, in the form of directed branches of the graph. Judea Pearl pointed out that causality is an extra information relating to random variables, not inferrable from their joint probability distribution [51]. Bayesian

networks also naturally support incremental execution. In this section, we show how these concepts supporting causality and incremental sampling, can be extended to support nondeterminism.

3.1 *Mixed Kernel and Mixed Bayesian Network*

In basic probability theory, a *transition probability* (also called *probability kernel*), is a map $y \mapsto P(x|y)$ that specifies the probability that a random variable X takes the value x knowing that another random variable Y takes the value y. This notion is extensively used in Bayesian reasoning.

We begin by extending the notion of probability kernel to that of Mixed Kernel. The starting idea consists in defining a Mixed Kernel as a function, mapping every Y-state of a set Y of variables, to a Mixed System having X as its set of variables. For the notations used in the sequel, the reader is referred to the beginning of Section 2. We shall denote by

$$\mathbb{S}(X) \tag{27}$$

the class of all (possibly inconsistent) Mixed Systems having X as their set of variables.

Definition 8 (Mixed Kernel) A *Mixed Kernel* (or simply *kernel*) is a map

$$K : Q_X \to \mathbb{S}(X')\,,$$

where X and X' are two finite sets of variables such that $X \cap X' = \emptyset$, called the sets of *inputs* and *outputs* of kernel K. In the sequel, we shall denote these two sets X and X' by X_K^{in} and X_K^{out}, respectively.

The *probabilistic semantics* of K is the map

$$q_{\text{in}} \mapsto \overline{\pi}(q_{\text{in}}) \tag{28}$$

where q_{in} is a value for the input variables X_K^{in} and $\overline{\pi}(q_{\text{in}})$ is the outer probability associated to Mixed System $K(q_{\text{in}})$—by (14) there is no need to consider $\underline{\pi}(q_{\text{in}})$. □

For $q{\in}Q, \omega{\in}\Omega$ and $C \subseteq \Omega{\times}Q$, we write

$$C_q \;=_{\text{def}}\; \{\omega{\in}\Omega \mid \omega C q\},\;\text{ and }\; C_\omega \;=_{\text{def}}\; \{q{\in}Q \mid \omega C q\}\,. \tag{29}$$

Convention 1 A Mixed Kernel K whose input set X is empty identifies with the Mixed System $S = K(\epsilon)$ it defines, where Q_X is the singleton $\{\epsilon\}$ (the distinguished state ϵ was introduced in (25)). Vice-versa, any system S identifies with the Mixed Kernel K whose input set X is empty and $K(\epsilon) = S$. □

Definition 9 (Mixed Bayesian Network) Let $\mathcal{N}=(X \cup \mathbb{K}, \hookrightarrow)$ be a directed acyclic bipartite graph, where X and $\mathbb{K}$ are finite sets of variables and Mixed Kernels, and $\hookrightarrow \subseteq (X \times \mathbb{K}) \cup (\mathbb{K} \times X)$ is the set of edges. For $K \in \mathbb{K}$, we denote by ${}^\bullet K$ and $K^\bullet$ the sets of variables $x \in X$ such that $x \hookrightarrow K$ and $K \hookrightarrow x$, respectively. $\mathcal{N}$ is called a *Mixed Bayesian Network* if satisfies the following conditions:

$$\forall K \in \mathbb{K} \implies X_K^{\text{in}} \subseteq {}^\bullet K \text{ and } X_K^{\text{out}} = K^\bullet . \tag{30}$$

$$\forall K_1, K_2 \in \mathbb{K}, K_1 \neq K_2 \implies K_1^\bullet \cap K_2^\bullet = \emptyset \tag{31}$$

For convenience, we will denote by

$$K_1 ; K_2 \tag{32}$$

a Bayesian network $\mathcal{N}=(X \cup \mathbb{K}, \hookrightarrow)$ whose set $\mathbb{K}$ contains only two Mixed Kernels K_1 and K_2, such that $K_1 \hookrightarrow K_2$ and $X = X_{K_1}^{\text{in}} \cup X_{K_1}^{\text{out}} \cup X_{K_2}^{\text{in}} \cup X_{K_2}^{\text{out}}$. □

This notion is illustrated on Figure 4 for two Mixed Kernels communicating via variable x (compare with Figure 3).

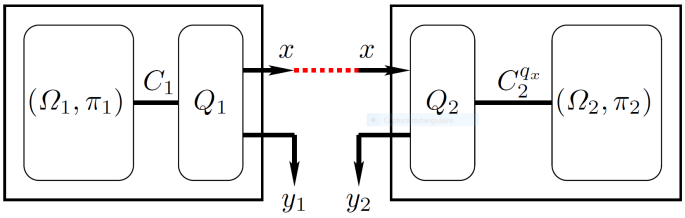


Fig. 4 Bayesian Network $S_1 ; K_2$. Mixed Kernel K_2 has input x.

Example 11 (Mixed Bayesian Network from running example) This example is a follow-up of Example 9, whose notations are reused. Consider again the running example (6,7), in which we discard equations (e_0) and (e_2), and rewrite it for convenience as shown in Figure 5, left. In the same figure, we show the

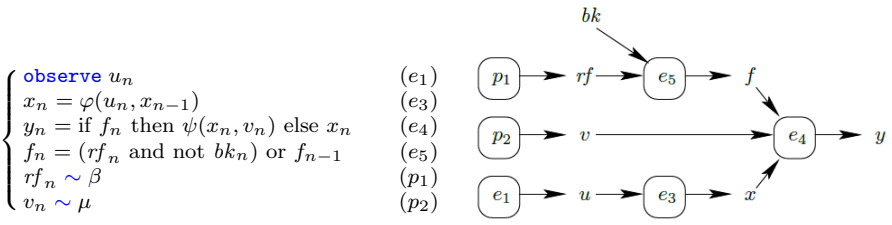


Fig. 5 Left: a subsystem of running example (6,7); right: the associated dataflow graph (time indices were discarded).

dataflow graph defined by this subsystem on the right: it is a directed bipartite graph whose vertices are the variables and the equations. A directed branch (e, x) exists if variable x is determined by equation e, and a directed branch (u, e) exists if u is a free variable in equation e. The directed graph shown is acyclic. To regard this diagram as a Bayesian network following Definition 9, it remains to map every equation vertex to a Mixed Kernel possessing the due properties. Consider box p_1, representing the statement $rf \sim \beta$; p_1 is a probability distribution producing rf, which is a Mixed Kernel by Convention 1. Consider then box e_4, representing the statement $y = \text{if } f \text{ then } \psi(x, v) \text{ else } x$; it is a Mixed Kernel having f, x, v as inputs and producing y via a pure nondeterministic Mixed System once the input values are given.

We could also remove the directions of the arrows from the above dataflow graph, thus regarding it as a factor graph, representing the parallel composition $S_{(p_1)} \parallel S_{(p_2)} \parallel S_{(e_1)} \parallel S_{(e_3)} \parallel S_{(e_4)} \parallel S_{(e_5)}$. A natural question is to compare the above two alternative semantics given to the same model, via Mixed Bayesian Networks and via Mixed Systems, respectively. This will be addressed in Theorem 4 and illustrated with Example 15. □

We associate a Mixed Bayesian Network $\mathcal{N}{=}(X{\cup}\mathbb{K}, \hookrightarrow)$ with the partial order $(X{\cup}\mathbb{K}, \preceq)$, where $\preceq$ is the transitive closure of $\hookrightarrow$. Let q be a valuation of the set X of variables and K a kernel belonging to $\mathbb{K}$, we write $q_{\bullet K}$ and $q_{K\bullet}$ for the restriction of q to the variables belonging to ${}^\bullet K$ and $K^\bullet$, respectively.

Definition 10 (incremental sampling and probabilistic semantics) The *incremental sampling* of a Mixed Bayesian Network $\mathcal{N}$ is defined by structural induction over $\preceq$ as follows:

1. Initial condition: we assume a value for every variable $x \in \min(X \cup \mathbb{K})$, where min refers to $\preceq$; we set $X_- = \min(X \cup \mathbb{K}) \cap X$ and $\mathbb{K}_- = \emptyset$;
2. Induction hypothesis: $X_- \cup \mathbb{K}_- \subseteq X \cup \mathbb{K}$ is a downward closed subset of vertices of $\mathcal{N}$ such that
   (a) $\mathbb{K}_-^\bullet \subseteq X_-$;
   (b) every variable $x \in X_-$ holds a value, whereas every $x \notin X_-$ does not;
3. Induction step: while $X_- {\neq} X$, do:
   (a) let $\mathbb{K}^* \subseteq \mathbb{K}{-}\mathbb{K}_-$ collect the kernels K such that ${}^\bullet K{\subseteq}X_-$ and $K^\bullet{\neq}\emptyset$;
   (b) for every $K{\in}\mathbb{K}^*$, every variable belonging to ${}^\bullet K$ holds a value, hence we can sample Mixed System $K(q_{\bullet K})$, which returns a value for $q_{K\bullet}$;
   (c) doing this for all $K{\in}\mathbb{K}^*$ yields a value for every variable belonging to $X_- \cup \mathbb{K}^{*\bullet} \supset X_-$ (the inclusion is strict);
   (d) set $\mathbb{K}_- := \mathbb{K}_- \cup \mathbb{K}^*$ and $X_- := X_- \cup \mathbb{K}^{*\bullet}$ and return to 3.
4. Done.

Sampling $\mathcal{N}$ thus returns a value $q \in Q_X$ for every variable belonging to X, we denote this by $\mathcal{N} \rightsquigarrow q$. The *probabilistic semantics* of $\mathcal{N}$ is the map $q \mapsto \overline{\pi}(q)$, associating every $q{\in}Q_X$ such that $\mathcal{N} \rightsquigarrow q$, with its *probabilistic score*

$$\overline{\pi}(q) = \prod_{K \in \mathbb{K}} \overline{\pi}\big(K, q_{\bullet K}\big)\big(q_{K\bullet}\big)\,. \tag{33}$$

In (33), $\overline{\pi}(K, q_{\bullet K})(q_{K\bullet})$ is the score assigned to state $q_{K\bullet}$ by the outer probability associated with the mixed system $K(q_{\bullet K})$. □

Since inclusion $X_- \cup \mathbb{K}^{*\bullet} \supset X_-$ in step 3c is strict, the inductive procedure terminates in finitely many steps. The inductive procedure of Definition 10 is formalized in Algorithm 1.

Algorithm 1 Incremental sampling of Mixed Bayesian Network $\mathcal{N}$

Require: $\forall x \in \min(X \cup \mathbb{K}),\ x$ is defined
Ensure: $\forall x \in X,\ x$ is defined
  $X_- \leftarrow X \cap \min(X \cup \mathbb{K})$ and $\mathbb{K}_- \leftarrow \emptyset$
  while $X_- \neq X$ **do**
    $\mathbb{K}^* \leftarrow \{\ K\ \mid\ {}^\bullet K \subseteq X_-$ and $K^\bullet \neq \emptyset\ \}$
    for all $K \in \mathbb{K}^*$ **do**
      sample$(K(q_{\bullet K}))$
    end for
  end while
  $\mathbb{K}_- \leftarrow \mathbb{K}_- \cup \mathbb{K}^*$
  $X_- \leftarrow X_- \cup \mathbb{K}^{*\bullet}$

Definition 11 (Mixed Bayesian Network equivalence) Let $\mathcal{N}_1$ and $\mathcal{N}_2$ be two Mixed Bayesian Networks such that $X_1 = X_2$. Say that $\mathcal{N}_1$ and $\mathcal{N}_2$ are *probabilistically equivalent,* written $\mathcal{N}_1 \equiv_P \mathcal{N}_2$, if they possess equal probabilistic semantics: $\overline{\pi}_1 = \overline{\pi}_2$. □

By Lemma 1, $S \equiv S'$ implies $S \equiv_P S'$, when regarding mixed systems S and S' as Mixed Bayesian Networks.

Example 12 (Finite Markov chain as a Mixed Bayesian Network) Recall that a finite sequence of random variables $X_1, X_2, \ldots, X_n$ is called a Markov chain if the joint distribution of $(X_0, X_1, \ldots, X_n)$ factorizes as $\pi(X_0{=}x_0, \ldots, X_n{=}x_n) = \mu(x_0) \prod_{i=1}^{n} P(x_i|x_{i-1})$, where the probability μ over $\mathbf{X}$, the state space of the Markov chain, is the *initial condition* and $P(x'|x)$ is the *transition kernel,* i.e., for x fixed, $x' \mapsto P(x'|x)$ is a probability over x'. Markov chains are thus a particular case of the Mixed Bayesian Networks proposed in Definition 9. □

Theorem 1 (composing Mixed Bayesian Networks) *Let $\mathcal{N}_1$ and $\mathcal{N}_2$ be two Mixed Bayesian Networks such that $\mathbb{K}_1 \cap \mathbb{K}_2 = \emptyset$. The union $\mathcal{N} =_{\text{def}} \mathcal{N}_1 \cup \mathcal{N}_2$ of the two directed graphs yields a Mixed Bayesian Network if the following conditions hold:*

1. *Directed graph $\mathcal{N}_1 \cup \mathcal{N}_2$ is acyclic; and*
2. *Condition (31) of Definition 9 is satisfied.*

The two graphs interact via their shared variables $X_1 \cap X_2$.

Proof Condition(30) of Definition 9 is local to the Mixed Kernels, and remains satisfied when taking the union of the graphs. Thus the satisfaction of Conditions 1 and 2 implies that all conditions of Definition 9 are satisfied. □

3.2 *Bayesian calculus: relating Mixed Bayesian Networks and Mixed Systems*

As a preamble, we recall some facts from basic probability theory. For a pair (X, Y) of random variables with joint distribution $P(x, y)$, usual Bayes formula writes $P(x, y)=P(y)P(x|y)$, where $P(y) =_{\text{def}} \sum_x P(x, y)$ is the *marginal distribution* of Y and $P(x|y)$ is the *conditional distribution* of X given that $Y=y$, assigning, to each value y of Y, a probability for X. In this section, we extend Bayesian reasoning to the blending of probability and nondeterminism, and we relate the two models of Mixed Bayesian Networks and Mixed Systems.

As a first step, we extend the notion of conditional distribution to mixed systems. We will use the following notation: Given a set of variables Y and $q_Y \in Q_Y$, we write

$$(Y{=}q_Y) \tag{34}$$

denotes the Mixed System defined as follows: Ω is the singleton $\{1\}$ with trivial probability on it, Y is the set of variables, and $C = \{(1, q_Y)\}$ is a singleton, expressing that Y is constrained to take the value q_Y.

Definition 12 (conditional) Let $S = (\Omega, \pi, X, C)$ be a Mixed System, and let $Y \subseteq X$ be a subset of variables. The *conditional* of S on Y, denoted by $\mathbf{Cond}_Y(S)$, is the kernel defined by $\mathbf{Cond}_Y(S)\,(q_Y) =_{\text{def}} (Y{=}q_Y) \parallel S$. □

Example 13 (Link with the classical notion) Consider the following particular case for S: $\Omega{=}Q$, and C is the diagonal of $\Omega{\times}Q$. Then, S specifies the joint distribution π for set X of random variables. Decompose $X = Y{\cup}Z$ where $Y{\cap}Z = \emptyset$. Compressing $\mathbf{Margin}_Y(S)$ yields the marginal distribution of Y. Compressing $(Y{=}q_Y) \parallel S$ yields the conditional distribution $\pi(q_Z|q_Y)$. Therefore, Definitions 6 and 12 extend the notions of marginal and conditional existing on purely probabilistic systems. □

Generally, sampling the parallel composition $S_1 \parallel S_2$ yields a result that differs from the incremental sampling of $S_1; \mathbf{Cond}_{X_1}(S_2)$ (by Convention 1 we can regard S_1 as a kernel and consider this incremental sampling). Nevertheless, the following result holds (see Definition 11 regarding $\equiv_P$):

Theorem 2 (Bayes formula) *Let $S = (\Omega, \pi, X, C)$ be a Mixed System and $Y \subseteq X$ a subset of variables. Then, the following* Bayes formula *holds:*[3]

$$S \equiv_P \mathbf{Margin}_Y(S)\,;\mathbf{Cond}_Y(S)\ .$$

[3] This theorem and formula (12) correct the erroneous construction of the conditional $\mathbf{Cond}_Y(S)$ in Appendix A of [13].

Proof See Appendix B.1 for the proof. □

The following corollary of Bayes formula is needed in the proof of Theorem 3.

Corollary 1 *Let S_1, S_2 be any two Mixed Systems, and let Y be a set of variables containing $X_1 \cap X_2$. Then:*

$$S_1 \parallel S_2 \equiv_P \big(S_1 \parallel \mathbf{Margin}_Y(S_2)\big); \mathbf{Cond}_Y(S_2) \ . \tag{35}$$

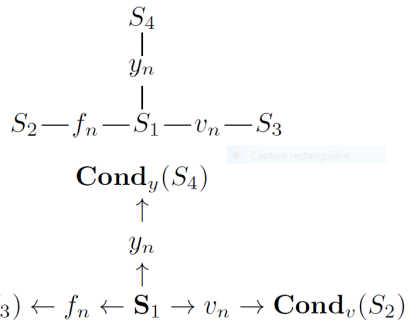
Proof See Appendix B.2. □

By Definition 7, the parallel composition $\mathcal{S} = \prod_{S \in \mathbb{S}} S$ defines a *Factor Graph* $\mathcal{G}_\mathcal{S}$, having nondirected bipartite edges $S - x$, for every $S \in \mathbb{S}$ and every visible variable x of S. Message passing algorithms transform certain Factor Graphs associated with a parallel composition of several Mixed Systems, to Mixed Bayesian Networks while preserving probabilistic equivalence. This provides such Factor Graphs with an incremental sampling:

Theorem 3 (message passing algorithm) *If the Factor Graph $\mathcal{G}_\mathcal{S}$ of system $\mathcal{S}$ is a tree, we can transform it into a Mixed Bayesian Network $\mathcal{N}_\mathcal{S}$ while preserving its probabilistic semantics.*

See Appendix B.3 for a proof.

Example 14 (running example, message passing) The following picture displays, on the top, the Factor Graph associated with S_1 and S_2 and S_3 and S_4, and, on the bottom, the Mixed Bayesian Network resulting from applying the message passing algorithm—for better readability we show only shared variables:



where $\mathbf{S}_1 =_{\text{def}} S_1 \parallel \mathbf{Margin}_v(S_2) \parallel \mathbf{Margin}_f(S_3) \parallel \mathbf{Margin}_y(S_4)$. □

In the second part of this section we further study the link between the two models of Mixed Systems and Mixed Bayesian Networks. We begin with a key lemma:

Corollary 2 *Let S_1, S_2 be any two Mixed Systems, and let Y be a set of variables containing $X_1 \cap X_2$. If $S_1 \parallel \mathbf{Margin}_Y(S_2) \equiv S_1$ holds, then it follows that*

$$S_1 \parallel S_2 \ \equiv_P \ S_1; \mathbf{Cond}_Y(S_2) \ . \tag{36}$$

Condition $S_1 \parallel \mathbf{Margin}_Y(S_2) \equiv S_1$ expresses that composing S_2 with S_1 does not affect S_1. That is, when sampling the parallel composition $S_1 \parallel S_2$, we can first sample S_1, which in turn affects the sampling of S_2. Equivalently, $S_1 \parallel S_2$ can be seen as a dataflow network in which S_1 executes first. How the sequential execution proceeds is formalized by the right-hand side of (36).

Proof Direct consequence of Corollary 1. □

To compare the two models of Mixed Systems and Mixed Bayesian Networks, we first need to define an embedding, from the former into the latter. We begin with Mixed Kernels. To each Mixed Kernel

$$K : Q_Y \to \mathbb{S}(Y') \,, \text{ where } K(q_Y) = \left(\Omega_{q_Y}, \pi_{q_Y}, Y', C_{q_Y}\right) ,$$

we associate the Mixed System $S_K = (\Omega, \pi, X, C)$, defined as follows:

$$\begin{aligned} &\Omega = \textstyle\prod_{q_Y \in Q_Y} \Omega_{q_Y} \;;\; \pi = \textstyle\prod_{q_Y \in Q_Y} \pi_{q_Y} \;;\; X = Y \cup Y' \\ &C = \left\{ \left(\omega, (q_Y, q_{Y'})\right) \,\middle|\, \omega_{q_Y} \, C_{q_Y} \, q_{Y'} \right\} \end{aligned} \tag{37}$$

This mapping is constructed by reusing the method of Example 3: the variables of the Mixed System are the union of inputs and outputs of the Mixed Kernel; the probability space is the product of all probability spaces of the targets of the Mixed Kernel, i.e., the Ω_{q_Y}'s; and the relation C_{q_Y} of $K(q_Y)$ is satisfied by $q_{Y'}$. Then, to any Mixed Bayesian Network $\mathcal{N} = (X \cup \mathbb{K}, \hookrightarrow)$, we associate the Mixed System

$$S_{\mathcal{N}} \;=\; \|_{K \in \mathbb{K}} \, S_K \,. \tag{38}$$

The following theorem relates $S_{\mathcal{N}}$ to $\mathcal{N}$:

Theorem 4 *The following conditions ensure that $S_{\mathcal{N}}$ and $\mathcal{N}$ are probabilistically equivalent: $S_{\mathcal{N}} \equiv_P \mathcal{N}$:*

1. *All the minimal vertices of $\mathcal{N}$ are Mixed Systems, and*
2. *For every $K \in \mathbb{K}$ and every $q \in {}^\bullet K$, Mixed System $K(q)$ is consistent.*

Condition 1 ensures that $\mathcal{N}$ is free from nondeterministic input variables, and Condition 2 guarantees that every $K \in \mathbb{K}$ is non-blocking.

Proof See Appendix B.4. □

Example 15 (follow-up of Example 11) We consider the example of Figure 5 again. Since the considered example can be given a dataflow graph, Theorem 4 applies, showing the probabilistic equivalence between the two semantics. In Section 6 regarding the semantics of ReactiveBayes mini language, the above argument is made systematic in Theorem 7. □

Thus, Theorem 4 can be interpreted as follows: if a parallel composition of Mixed Systems can be seen as a dataflow graph, then it can be executed incrementally.

3.3 *Discussion*

Discussion 5 (computing generalized likelihoods) The purpose of probabilistic languages [44,19,34] is not only (actually, not so much) sampling but rather estimation/inference. Of course, in addition to performing incremental sampling, Bayes' formula also allows evaluating probabilities of properties incrementally. Then, a counterpart of Bayes' formula exists for performing maximum likelihood estimation incrementally—it is known in the pattern recognition literature as the Viterbi algorithm [29,54]. Theorem 3 shows that message passing algorithms also allow for an incremental evaluation of generalized likelihoods.

So far, we have presented models involving no dynamics. In the next section, we move to our proposed formal model for dynamical systems: Mixed Automata.

4 Mixed Automata

The idea is the following: we upgrade notions, from automata, to Probabilistic Automata, and to Mixed Automata:

1. Transitions $q \xrightarrow{\alpha} q'$, where q and q' are states and α is an action, correspond to automata.
2. Upgrading them to $q \xrightarrow{\alpha} \pi' \rightsquigarrow q'$, where π' is the next probabilistic state and $\rightsquigarrow$ denotes probabilistic sampling, yields Simple Probabilistic Automata following Segala and Lynch [56,45].
3. Upgrading them further to $q \xrightarrow{\alpha} S' \rightsquigarrow q'$, where S' is a Mixed System and $\rightsquigarrow$ denotes sampling, yields Mixed Automata.

4.1 *The model of Mixed Automata*

The formal definition is introduced next. It uses the notation $\mathbb{S}(X)$, introduced in (27). We assume an underlying alphabet $\mathbf{\Sigma}$ of *actions*.

Definition 13 (Mixed Automaton) A *Mixed Automaton* is a tuple

$$M = (\Sigma, X, q_0, \rightarrow),$$

where: $\Sigma \subseteq \mathbf{\Sigma}$ is a finite set of actions, X is a finite set of variables having domain $Q = \prod_{x \in X} Q_x$, $q_0 \in Q$ is the *initial* state, and $\rightarrow \subseteq Q \times \Sigma \times \mathbb{S}(X)$ is the *transition relation*. We write

$$q \xrightarrow{\alpha} S \text{ (or } q \xrightarrow{\alpha}_M S \text{ when we wish to make } M \text{ explicit)}$$

to mean $(q, \alpha, S) \in \rightarrow$. We require M to be *deterministic*:

$$\text{for any pair } (q, \alpha) \in Q \times \Sigma,\ q \xrightarrow{\alpha} S \text{ and } q \xrightarrow{\alpha} S' \text{ implies } S = S'. \qquad (39)$$

The *sampling* of M is its set of *runs* $\mathbf{r}$, which are finite sequences of chained transitions:

$$\mathbf{r} = q_0 \xrightarrow{\alpha_1} S_1 \rightsquigarrow q_1 \xrightarrow{\alpha_2} S_2 \rightsquigarrow q_2 \dots q_{k-1} \xrightarrow{\alpha_k} S_k \rightsquigarrow q_k \,, \tag{40}$$

where Mixed Systems $S_1, \dots, S_k$ are consistent, and $S \rightsquigarrow q$ is the sampling introduced in Definition 1. □

The transitions of Mixed Automata target Mixed Systems, which combine nondeterminism with probabilities. Therefore, Mixed Automata capture nondeterminism despite Condition (39).

Example 16 (comparing with classical notions) Let $(X_n)_{n\geq 0}$ be a Markov chain with state space Q, initial state q_0, and transition probability $P(q' \,|\, q)$. We can reformulate it as the Mixed Automaton $M = (\Sigma, X, q_0, \rightarrow)$, where: Σ is the singleton $\{\alpha\}$; variable X has domain Q; $\rightarrow$ maps (q, α) to the purely probabilistic Mixed System of Example 2, representing probability $q' \mapsto P(q' \,|\, q)$ for given state q. □

Like automata and Probabilistic Automata, Mixed Automata come equipped with a notion of parallel composition, built on top of the parallel composition of Mixed Systems. The simplest idea is that the transitions of parallel composition $M_1 \parallel M_2$ will take the form $q_1 \sqcup q_2 \xrightarrow{\alpha} S_1' \parallel S_2' \rightsquigarrow q_1' \sqcup q_2'$, where $q_1' \sqcup q_2'$ and $S_1' \parallel S_2'$ are defined in (22) and Definition 7, respectively. In this simple construction, synchronizing the two transitions is by having them perform the same action α.

To be able to define the semantics of our ReactiveBayes mini language, we will, however, need the more flexible synchronization mechanism of "compatible actions", provided by the standard synchronization algebra of process calculi, see, e.g. [16]. We thus assume that the underlying alphabet $\boldsymbol{\Sigma}$ of actions is equipped with a commutative and associative *join* partial operation $\sqcup_{\boldsymbol{\Sigma}} : \boldsymbol{\Sigma} \times \boldsymbol{\Sigma} \to \boldsymbol{\Sigma}$, where $\alpha_1 \sqcup_{\boldsymbol{\Sigma}} \alpha_2$ is defined whenever the two actions are *compatible,* written $\alpha_1 \bowtie_{\boldsymbol{\Sigma}} \alpha_2$. In the composition of Mixed Automata, the components synchronize on compatible actions and move to the parallel composition of target systems by performing the join of the two actions:

Definition 14 (parallel composition) Let M_1 and M_2 be two Mixed Automata having compatible initial states $q_{0,1} \bowtie q_{0,2}$. Their *parallel composition* $M_1 \parallel M_2$ has alphabet $\Sigma_1 \cup \Sigma_2$, set of variables $X_1 \cup X_2$, and initial state $q_{0,1} \sqcup q_{0,2}$. Its transition relation $\longrightarrow_M$ is the minimal relation satisfying the following condition, where $S_1 \parallel S_2$ was defined in Definition 7:

$$\left.\begin{array}{l} q_i \xrightarrow{\alpha_i}_{M_i} S_i \text{ for } i = 1, 2 \\ q_1 \bowtie q_2 \quad \text{and} \quad \alpha_1 \bowtie_{\boldsymbol{\Sigma}} \alpha_2 \end{array}\right\} \implies q_1 \sqcup q_2 \xrightarrow{\alpha}_M S_1 \parallel S_2, \text{ where } \alpha = \alpha_1 \sqcup_{\boldsymbol{\Sigma}} \alpha_2 \,. \; \square$$

The next important notion is (bi)simulation, which is central to automata theory. We upgrade it, from the basic notion for automata to the extended notion for Mixed Automata:

1. In the context of automata, a relation $\leq$ on pairs of states is a *simulation* if it satisfies [55]:

$$\left.\begin{array}{c} q_1 \xrightarrow{\alpha} q_1' \\ q_1 \leq q_2 \end{array}\right\} \implies \exists q_2' : \left\{\begin{array}{c} q_2 \xrightarrow{\alpha} q_2' \\ q_1' \leq q_2' \end{array}\right.$$

2. This definition is upgraded to Probabilistic Automata as follows [55]:

$$\left.\begin{array}{c} q_1 \xrightarrow{\alpha} \pi_1' \\ q_1 \leq q_2 \end{array}\right\} \implies \exists \pi_2' : \left\{\begin{array}{c} q_2 \xrightarrow{\alpha} \pi_2' \\ \pi_1' \leq^P \pi_2' \end{array}\right.$$

   where $\leq^P$ is the *lifting of* $\leq$ *to pairs of probabilistic states.* We have:

$$\begin{array}{l} \pi_1' \leq^P \pi_2' \text{ ensures, for each } q_1' \text{ such that } \pi_1' \rightsquigarrow q_1', \\ \text{the existence of } q_2' \text{ satisfying } \pi_2' \rightsquigarrow q_2' \text{ and } q_1' \leq q_2'. \end{array} \tag{41}$$

3. This definition will be further upgraded to Mixed Automata as follows:

$$\left.\begin{array}{c} q_1 \xrightarrow{\alpha} S_1' \\ q_1 \leq q_2 \end{array}\right\} \implies \exists S_2' : \left\{\begin{array}{c} q_2 \xrightarrow{\alpha} S_2' \\ S_1' \leq^S S_2' \end{array}\right. \tag{42}$$

   where $\leq^S$ is the *lifting of* $\leq$ *to pairs of Mixed Systems.* We request:

$$\begin{array}{l} S_1' \leq^S S_2' \text{ shall ensure, for each } q_1' \text{ such that } S_1' \rightsquigarrow q_1', \\ \text{the existence of } q_2' \text{ satisfying } S_2' \rightsquigarrow q_2' \text{ and } q_1' \leq q_2'. \end{array} \tag{43}$$

Such a lifting is introduced next. Let S_1 and S_2 be two Mixed Systems.

Definition 15 (lifting relations on Mixed Systems states) Let $\rho \subseteq Q_1 \times Q_2$ be any state relation. Mixed System relation $\rho^{\mathbb{S}} \subseteq \mathbb{S}(X_1) \times \mathbb{S}(X_2)$ is the *lifting* of ρ if there exists a *weighting* function $w : \Omega_1 \times \Omega_2 \to [0,1]$ such that:

1. For every triple $(\omega_1, \omega_2, q_1) \in \Omega_1 \times \Omega_2 \times Q_1$ such that $w(\omega_1, \omega_2) > 0$ and $\omega_1 \, C_1 \, q_1$, there exists $q_2 \in Q_2$ such that $q_1 \, \rho \, q_2$, and $\omega_2 \, C_2 \, q_2$;
2. Weighting w projects to π_1 and π_2:

$$\sum_{\omega_2} w(\omega_1, \omega_2) = \pi_1(\omega_1) \text{ and } \sum_{\omega_1} w(\omega_1, \omega_2) = \pi_2(\omega_2). \qquad \square$$

By construction, this definition for the lifting of state relations to relations on Mixed Systems satisfies (43). Note the existential quantifier in Condition 1. By Condition 2, w induces a probability on $\Omega_1 \times \Omega_2$. We write $S_1 \, \rho^{\mathbb{S}} \, S_2$ to mean $(S_1, S_2) \in \rho^{\mathbb{S}}$.

Lemma 3 *$S_1 \, \rho^{\mathbb{S}} \, S_2$ and $S_1' \equiv S_1$ together imply $S_1' \, \rho^{\mathbb{S}} \, S_2$.*

See Appendix C.1 for a proof. $\square$

Definition 16 (simulation) Given two Mixed Automata M_1, M_2, we say that M_2 *simulates* M_1, written $M_1 \leq M_2$, if they possess a *simulation,* i.e., a relation $\leq \subseteq Q_1 \times Q_2$ such that $q_{0,1} \leq q_{0,2}$ and, for every pair $q_1 \leq q_2$ and every transition $q_1 \xrightarrow{\alpha}_1 S_1$, there exists a transition $q_2 \xrightarrow{\alpha}_2 S_2$ such that $S_1 \leq^{\mathbb{S}} S_2$, where $\leq^{\mathbb{S}}$ denotes the lifting of $\leq$. M_1 and M_2 are called *simulation equivalent* if they simulate each other. M_1 and M_2 are called *bisimilar* if there exists a relation $\sim \subseteq Q_1 \times Q_2$ such that both $\sim$ and its transpose are simulations. □

The notion of simulation and its derived constructs are the core topic of the literature on automata and their probabilistic extensions. The reader is referred to the next section for a bibliographical discussion.

Lemma 4 *Parallel composition preserves simulation:* $M_1' \leq M_1$ *and* $M_2' \leq M_2$ *together imply* $M_1' \parallel M_2' \leq M_1 \parallel M_2$.

See Appendix C.2 for a proof. □

4.2 *Bibliographical discussions*

We conclude this section with some bibliographical discussions.

Discussion 6 (lifting and coupling) Our lifting is a direct extension of the technique used in [55] for Probabilistic Automata. In the context of probabilistic reasoning, the same technique was also extensively studied under the name of *probabilistic coupling* [8,36]. Weighting function $w(\omega_1, \omega_2)$ of Definition 15 transposes probabilistic coupling to our model of Mixed Automata in which nondeterminism and probability are combined. In a different community, "stochastic nondeterminism" was extensively studied through the notion of *nondeterministic labeled Markov process* in [22,27], in a categorical framework; the second reference encompasses continuous distributions (beyond discrete).

Discussion 7 (simulation equivalence vs. bisimilarity) Despite the condition (39) that the transition relation shall be deterministic, the two notions of "'simulation equivalence" and "bisimilarity" differ. The reason is that nondeterminism is hidden behind the Mixed Systems targeted by transitions. In our forthcoming Theorem 6, we will prove that Segala's Probabilistic Automata [55,56,45], which possess nondeterministic transition relations, can be embedded into Mixed Automata while preserving simulations.

Discussion 8 (Mixed Automata are causal in time) Mixed Automata remain a *causal* model in time since the current transition depends on the past, not on the future. Consequently, Mixed Automata cannot be used to specify acausal estimation problems, e.g., estimating unmeasured variable z_k based on observations of $X_0, \ldots, X_k, \ldots, X_N$. To perform this, we must "unfold time as space", i.e., regard $X_0, \ldots, X_N$ as a $(N+1)$-set of variables, not as successive occurrences in time of variable X. Note that the transition relations of Mixed Automata inherit, from Mixed Systems, the Bayesian Calculus and the notions of Factor Graph and Mixed Bayesian Network.

5 Comparison with Segala's Probabilistic Automata

Segala and Lynch originally proposed Probabilistic Automata (PA) [55,56,45]. To simplify our comparison, we discuss the version of PA without internal actions. According to the classification made by Sokolova and de Vink [58], we study the link with both the Simple (Segala) Probabilistic Automata and the (Segala) Probabilistic Automata. For the former, actions are selected, and then a transition to a probabilistic state is selected nondeterministically. For the latter, both the action and a state are jointly selected, probabilistically. This distinction is referred to as reactive vs. generative models in [58].

Simple Probabilistic Automata existed way before the work of Segala and Lynch [55,56,45], in the community of applied mathematics and probability theory, where they are known under the name of *Markov Decision Processes (MDP)* [10,53]. In this context, the main considered problem is the synthesis of an *optimal policy* to minimize some expected cost function on the trajectories of the system. The minimization is over *scheduling policies,* which are causal rules for selecting the next action given the past trajectory. Once this policy has been fixed, the resulting dynamics is a Markov Chain. Studies on (bi)simulation were more recently developed for MDP's [31], and further developed to support robustness by defining metrics between finite MDP's [28]. In the following, $\mathcal{P}(Q)$ denotes the set of all probability distributions over the set Q. Formally, we consider a tuple $P = (\Sigma, Q, q_0, \rightarrow)$, where Σ is the finite alphabet of actions, Q is a finite state space, $q_0 \in Q$ is the initial state, and the probabilistic transition relation $\rightarrow$ is defined in two different ways:

$$\textit{Simple Probabilistic Automaton (SPA)} : \rightarrow \subseteq Q \times \Sigma \times \mathcal{P}(Q) \tag{44}$$

$$\textit{Probabilistic Automaton (PA)} : \rightarrow \subseteq Q \times \mathcal{P}(\Sigma \times Q) \tag{45}$$

In the following definitions, relation $\leq^{\mathcal{P}}$ is the lifting, to probability distributions over $Q \times Q'$, of the relation $\leq$ over $Q \times Q'$—for the definition of the lifting $\leq^{\mathcal{P}}$, the reader can use Definition 15 adapted by ignoring relations C_1 and C_2.

5.1 *Details for* SPA*, model* (44)

We write $q \xrightarrow{\alpha}_P \mu$ to mean $(q, \alpha, \mu) \in \rightarrow$ and $\mu \rightsquigarrow q'$ to mean that sampling μ returns next state q'. The sampling is: if P is in state $q \in Q$, performing $\alpha \in \Sigma$ leads to some target set of probability distributions over Q, of which one is selected, nondeterministically, and used to draw at random the next state q'. A *simulation relation* is a relation $\leq \subseteq Q \times Q'$ such that, for any $q \leq q'$, the following holds: if $q \xrightarrow{\alpha}_P \mu$, there exists μ' such that $q' \xrightarrow{\alpha}_{P'} \mu'$ and $\mu \leq^{\mathcal{P}} \mu'$. The *parallel composition* of SPA [45] is defined by: $P_1 \parallel P_2 = (\Sigma, Q, q_0, \rightarrow)$, where $\Sigma = \Sigma_1 \cup \Sigma_2$, $Q = Q_1 \times Q_2$, $q_0 = (q_{0,1}, q_{0,2})$, and $(q_1, q_2) \xrightarrow{\alpha} \mu_1 \times \mu_2$ holds iff $q_i \xrightarrow{\alpha}_i \mu_i$ for $i = 1, 2$.

5.2 *Details for* PA*, model* (45)

We write $q \longrightarrow_P \mu$ to mean $(q,\mu) \in \rightarrow$ and $\mu \rightsquigarrow (\alpha, q')$ to mean that sampling μ jointly returns action α and next state q'. The sampling is: P being in state $q \in Q$ leads to some target set of probability distributions over $\Sigma \times Q$, of which one is selected, nondeterministically, and used to draw at random the next pair (α, q') of action and state. A *simulation relation* is a relation $\leq \subseteq Q \times Q'$ such that, for any $q \leq q'$, the following holds: if $q \longrightarrow_P \mu$, there exists μ' such that $q' \longrightarrow_{P'} \mu'$ and $\mu \leq^{\mathcal{P}} \mu'$.

The *parallel composition* $P = P_1 \parallel P_2$ faces the following difficulty: there is a conflict between (1) the probabilistic choice of actions α_1 and α_2 in each component, and (2) the synchronization constraint on the pair (α_1, α_2) possibly required by the parallel composition.

This difficulty does not exist if no synchronization constraint exists, e.g., if the composition of actions $\alpha = \alpha_1.\alpha_2$ is always defined. In this case, the parallel composition is straightforward: $(q_1, q_2) \longrightarrow_P \mu_1 \times \mu_2$ iff $q_i \longrightarrow_{P_i} \mu_i$ holds for $i = 1, 2$. This kind of parallel composition does not capture synchronization, however.

In contrast, if strong synchronization is imposed, e.g., by requiring that $\alpha_1 = \alpha_2$ whenever one of the two actions is shared by the two components—this is the policy followed in our model of Mixed Automata—, then the above conflict exists. This conflict is usually resolved by adding a probabilistic scheduling policy specified through an auxiliary probability distribution; see the detailed discussion in [58] and references therein. A typical approach to compose the two transitions $q_i \longrightarrow_{P_i} \mu_i \rightsquigarrow (\alpha_i, q_i'), i = 1, 2$ is the following:

- If synchronization constraint $\alpha_1 = \alpha_2 = \alpha$ happens to be satisfied, then the two transitions synchronize and (q_1, q_2) leads to $(\alpha, (q_1', q_2'))$ with probability $\mu_1(\alpha, q_1') \times \mu_2(\alpha, q_2')$.
- If both actions α_1 and α_2 are local $\alpha_i \notin \Sigma_1 \cap \Sigma_2, i = 1, 2$, then the synchronization constraint is not violated. However, since only one action is permitted at a time in PA, one between the two transitions must be elected while the other is freezed. This is achieved by tossing a (possibly biased) coin with parameter $\sigma \in (0, 1)$, so that (q_1, q_2) leads to $(\alpha_1, (q_1', q_2))$ with probability $\mu_1(\alpha, q_1') \times \sigma$ and (q_1, q_2) leads to $(\alpha_2, (q_1, q_2'))$ with probability $\mu_2(\alpha_2, q_2') \times (1 - \sigma)$.
- Other cases are forbidden.

Collecting the outcomes that are not forbidden results in a transition of the form $q \longrightarrow_P \bar{\mu} \rightsquigarrow (\alpha, q')$, where $\bar{\mu}$ is *unnormalized*. A subsequent normalization is performed to get the final definition $q \longrightarrow_P \mu \rightsquigarrow (\alpha, q')$ for the transitions of the parallel composition. The definition of this parallel composition thus requires specifying an additional probability distribution (the parameter σ of the biased coin). In other variants, the conflict is solved by introducing *schedulers* as additional probability distributions.

5.3 *Comparison results*

The following theorems relate SPA and PA to Mixed Automata (proofs are constructive).

Theorem 5 (SPA vs. Mixed Automata)

1. *There exists a mapping $P \mapsto M_P$, from* SPA *to Mixed Automata, preserving both simulation and parallel composition: $P_1 \leq P_2$ iff $M_{P_1} \leq M_{P_2}$, whereas $M_{P_1 \parallel P_2}$ and $M_{P_1} \parallel M_{P_2}$ are simulation equivalent.*
2. *There exists a reverse mapping $M \mapsto P_M$, from Mixed Automata to* SPA*, preserving simulation. No reverse mapping exists, however, that preserves parallel composition.*

See Appendices D.1.1 and D.1.2 for proofs of Statements 1 and 2 of this theorem. The two mappings $P \mapsto M_P$ and $M \mapsto P_M$ are not opposite, which makes it possible for the two statements not to contradict each other. The non-existence of a reverse mapping $M \mapsto P_M$ preserving parallel composition highlights that the difference in the parallel compositions, for SPAs vs. for Mixed Automata, is deep.

Theorem 6 (PA vs. Mixed Automata) *There exists a mapping $P \mapsto M_P$, from* PA *to Mixed Automata, preserving simulation. Parallel composition, however, is not preserved.*

See Appendix D.2 for a proof.

Due to Statement 2 of Theorem 5 and the existence of an embedding SPA$\to$PA [58] preserving simulation, a reverse mapping exists from Mixed Automata to PA.

In [58], it is proved that SPA can be embedded into PA, by simply "pushing" actions from occurring prior to probabilistic choice to being part of probabilistic choice (in which case alternatives to emitting action α sum up to probability 1). So, it seems unnecessary to study the embeddings SPA $\to$ Mixed Automata and PA $\to$ Mixed Automata separately since mapping the second one seems sufficient. This is, however, not a good idea since the two embeddings differ, in that parallel composition is preserved for SPA but not for PA.

5.4 *Bibiographical discussions*

We conclude this section with some bibliographical discussions.

Discussion 9 (who comes first: nondeterminism or probability?) The following question is often considered [63]: should nondeterminism be resolved *prior* to or *after* probabilistic sampling? Since the selection of the performed action followed by that of one probability from a subset of $\mathcal{P}(Q)$ (for SPAs), or the selection of one probability from a subset of $\mathcal{P}(\Sigma \times Q)$ (for PAs) is performed prior to probabilistic sampling, both SPA and PA models follow the first alternative. Our model of Mixed Automata follows a schizophrenic approach: first, external nondeterminism is resolved by synchronizing on actions,

then, a probabilistic choice is performed by the so reached Mixed System, and finally, internal nondeterminism is resolved—one can thus say that nondeterminism is resolved "first-and-last". As we have seen in Section 5.3, the main difference between our model and models from the PA family is not in this "prior vs. after" issue, but instead in our handling of conditioning and parallel composition.

Discussion 10 (More on spa/pa versus Mixed Automata) So far, Theorems 5 and 6 compare SPA/PA and Mixed Automata regarding the core notions of PA, namely simulation and parallel composition. Conditioning is not considered in PA theories—this indeed is the reason for them to have problems when handling synchronization in the parallel composition. Furthermore, we do not see how factor graphs can be reflected in PA theories. In contrast, these concepts are naturally supported by our model of Mixed Automata. In addition, our model offers the classical concepts of PA theories, namely simulation, and equivalence.

6 The ReactiveBayes mini language and its semantics

In this section, we use the models of Mixed Systems and Mixed Automata to specify the semantics of the mini language we informally introduced in the introduction for our running example. To make this precise, we formalize this informal language through the "ReactiveBayes" syntax presented hereafter.

To prevent decidability issues in constraint solving, domains of variables and random variables are all assumed finite. Finally, to simplify our presentation of the syntax, domains are omitted.

6.1 *Syntax of* ReactiveBayes

Here is the syntax, where `keywords` are highlighted in blue:

$$\begin{array}{l} e \ ::= c \mid x \mid (e,e) \mid op(e) \mid f(e) \mid \texttt{pre}\ x \mid \texttt{init}\ x = c \\ \mathbf{S} ::= x \sim P(e) \mid e = e \mid \texttt{observe}\ x \mid \ \mathbf{S}\ \texttt{and}\ \mathbf{S} \end{array} \tag{46}$$

- An expression e is a constant c, a variable x, an external operator application $op(e)$, a function application $f(e)$, or a delayed version $\texttt{pre}\ x$ for the variable x. Initial condition $\texttt{init}\ x = c$ is required whenever $\texttt{pre}\ x$ occurs in the program; it fixes the initial value for x.
- A program $\mathbf{S}$ is the declaration of a *prior distribution* $P(e)$ for variable x, thus making it random; distribution $P(e)$ has, optionally, parameters set by expression e, an *equation* $e = e$, the declaration that *variable* x *is actually observed*, or the parallel composition $\ \texttt{and}\ $. For each term P we assume a semantics denoted by π_P, which is a probability.

No provision is given by syntax (46) for writing equations relating systems. In particular, fixpoint equations $\mathbf{S} = \mathbf{S}'\ \texttt{and}\ \mathbf{S}$ cannot be expressed: ReactiveBayes

does not offer full recursion. However, statements `pre` and `init` provide a limited form of recursion, supporting dynamical systems. This will be made clear in Section 6.3, where the semantics of full ReactiveBayes will be given.

Example 17 (running example in ReactiveBayes*)* The following two ReactiveBayes programs are proposed to formalize S_1, described in (1):

```
S1 :    observe u
   and  init x = x0
   and  y = phi(u,pre x)
   and  x = if fail then psi(y,noise) else y
```

and S_2, described in (2):

```
S2 :    init noise = n0
   and  noise = chi(pre noise,w)
   and  w ~ mu
```

6.2 *Semantics of the static fragment of* ReactiveBayes

We now give the semantics of the static fragment of ReactiveBayes, namely ignoring in (46) the statements `pre` and `init`. $[\![\mathbf{S}]\!]$ denotes the semantics of ReactiveBayes program $\mathbf{S}$. Two semantic domains can be considered, for this static fragment, namely:

- Mixed Systems, and
- Mixed Bayesian Networks.

6.2.1 Using Mixed Systems as a semantic domain

The semantics of a program will thus consist of a Mixed System (Ω, π, X, C). In writing this semantics, we will use the following notational conventions for specifying relation $C \subseteq \Omega \times Q$:

$$\begin{aligned} e = e' \text{ is } &\{(\omega, q) \in \Omega \times Q \mid e(X/q) = e'(X/q)\} \\ x = \omega \text{ is } &\{(\omega, q) \in \Omega \times Q \mid q_x = \omega\} \end{aligned}$$

In the first line, e and e' denote expressions over a set X of variables, and $e(X/q)$ denotes the value taken by the expression when q is substituted for X in e. In the second line, $Q = Q_X$ for a set X of variables, $x \in X$, and q_x is the value of x in state q. With these conventions, the semantics is:

$$\begin{aligned} &(i) && [\![\texttt{observe}\ x]\!] = (\{\omega\}, \delta_\omega, \{x\}, x = q) \\ &(ii) && [\![x \sim P]\!] = (\Omega_x, \pi_P, \{x\}, x = \omega_x) \\ &(iii) && [\![x \sim P(e)]\!] = \left(\textstyle\prod_{q \in Q_e} \Omega_q, \prod_{q \in Q_e} \pi_q, \{x, e\}, \bigcup_{q \in Q_e} (e{=}q \cap x{=}\omega_q)\right) \quad (47) \\ &(iv) && [\![e = e']\!] = (\{\omega\}, \delta_\omega, \mathrm{vars}(e) \cup \mathrm{vars}(e'), e = e') \\ &(v) && [\![\mathbf{S}_1\ \texttt{and}\ \mathbf{S}_2]\!] = [\![\mathbf{S}_1]\!] \parallel [\![\mathbf{S}_2]\!] \end{aligned}$$

In (i), the probabilistic part is trivial ($\Omega = \{\omega\}$ is a singleton and $\pi = \delta_\omega$ is the Dirac probability at that singleton), and there is a single visible variable x whose value $q \in Q_x$ is given but left unspecified. In (ii), probability distribution P is fixed; the semantics consists of the probability space (Ω_x, π_P), where Ω_x is a private copy of the domain of x equipped with probability π_P and having generic element $\omega_x \in \Omega_x$; equation $x = \omega_x$ exposes ω_x for further interactions through x. Line (iv) defines the semantics of equations; "vars(e)" denotes the set of variables involved in expression e; the semantics has a trivial probabilistic part, as in (i). Line (v) makes this semantics structural.

Line (iii) of the semantics deserves more explanations. The Mixed System defining this semantics involves two variables: x and e (a name for the expression). Variable e has domain Q_e. For each value $q \in Q_e$, a copy of the domain Q_x for variable x is created and called Ω_q. Ω_q is equipped with probability π_q, which is the semantics of $P(e)$ when e takes the value q. Cartesian product $\prod_{q\in Q_e} \Omega_q$ is equipped with the Cartesian product $\prod_{q\in Q_e} \pi_q$. Thus, each ω_q is obtained by sampling, independently from each other, Ω_q with distribution π_q. Finally, the relation C expresses that $x = \omega_q$ holds when $e = q$. Equivalently, x is sampled using distribution π_q when expression e takes the value q, which is the intent of this statement. This semantics reuses the method (37,38) for mapping Mixed Bayesian Networks to Mixed Systems. This complex semantics should be compared with the semantics of the same statement in terms of Mixed Bayesian Networks.

6.2.2 Using Mixed Bayesian Networks as semantic domain

In the following formulas, $[\![\ \mathbf{S}\ ]\!]$ denotes the Mixed Bayesian Network defined by $\mathbf{S}$, when it exists. Conditions for the existence of $[\![\ \mathbf{S}\ ]\!]$ is that 1) equations are restricted to be assignments of expressions (see (48,iv)), and 2) a success condition for (48,v) to be applicable is required, see hereafter. The Mixed Bayesian Network semantics thus applies only to a fragment of ReactiveBayes.

According to the definition of Mixed Bayesian Networks, the semantics consists of a bipartite directed graph whose vertices are either variables or Mixed Kernels, and the directed branches are denoted by $\hookrightarrow$.

$$\begin{array}{lll}
(i) & [\![\ \texttt{observe}\ x\ ]\!] & = [\![\texttt{observe}\ x]\!] \hookrightarrow x \\
(ii) & [\![\ x \sim P\ ]\!] & = [\![x \sim P]\!] \hookrightarrow x \\
(iii) & [\![\ x \sim P(e)\ ]\!] & = \mathrm{vars}(e) \hookrightarrow [\![x \sim P(e/q)]\!] \hookrightarrow x \\
(iv) & [\![\ x = e\ ]\!] & = \mathrm{vars}(e) \hookrightarrow [\![x = e]\!] \hookrightarrow x \\
(v) & [\![\ \mathbf{S}_1\ \texttt{and}\ \mathbf{S}_2\ ]\!] & = [\![\ \mathbf{S}_1\ ]\!] \cup [\![\ \mathbf{S}_2\ ]\!]
\end{array} \tag{48}$$

In (i), $[\![\texttt{observe}\ x]\!]$ is the Mixed System that was specified in (47) as being the semantics of statement `observe` x. Thus, (i) specifies that this system causes the value of x. In (ii), $[\![x \sim P]\!]$ is the Mixed System that was specified in (47) as being the semantics of statement $x \sim P$; it causes the value of x.

In (iii), the bipartite graph specifies that the network has one kernel vertex and input and output variables. The input variables are the variables involved in expression e, and the output variable is x. Once expression e is evaluated to

$q \in Q_e$, q is substituted for e in $P(e)$ (which is specified by e/q), and the Mixed System semantics of $x \sim P(q)$ is given by (47,*ii*). This is much simpler and more concise than (47,*iii*). The reason is that the intuition of the statement is causal: first, e gets evaluated, and then, x is sampled. Thus, Mixed Bayesian Networks, being naturally causal, are appropriate. In contrast, the semantics of (47,*iii*) had to be noncausal, since the domain was that of Mixed Systems, not Mixed Bayesian Networks.

Consider (47,*i–iv*) and (48,*i–iv*), providing the Mixed System and Mixed Bayesian Network semantics of primitive statements of the static fragment of ReactiveBayes. We wish to relate (47,*i*) and (48,*i*), etc. Let $S_{(47,i)}$ and $K_{(48,i)}$ be the Mixed System and Mixed Kernel giving the semantics of statement (i) of ReactiveBayes in (47) and (48), respectively, with corresponding notations for the other primitive statements (ii)–(iv). Referring to the correspondence (37), we claim that

$$S_{(47,i)} \equiv_P K_{(48,i)} \tag{49}$$

with corresponding properties for statements (ii)–(iv). This is immediate for statements (i), (ii) and (iii). In contrast, statement (iv) is purely nondeterministic and possesses inputs; its semantics in (47) does not exhibit the product probability space occurring in (37). Instead a trivial probability space $(\{\omega\}, \delta_\omega)$ is specified. But products of such trivial probability spaces remain isomorphic to $(\{\omega\}, \delta_\omega)$. So, our claim holds also for statement (iv).

Now, by Theorem 4, (49) implies that the two semantics are probabilistically equivalent.

To compare the two semantics for programs, we now consider (v). Not all well formed ReactiveBayes programs can be given a semantics in terms of Mixed Bayesian Networks. Indeed, the application of Rule (v) in (48) is subject to the following success condition (whose intuition is the dataflow diagram condition in Example 11):

Condition 1 (success condition for the existence of $[\![\mathbf{S}_1 \texttt{ and } \mathbf{S}_2]\!]$) *The union of the two directed acyclic graphs* $[\![\mathbf{S}_1]\!] \cup [\![\mathbf{S}_2]\!]$ *possesses no circuit and satisfies condition* (31) *of Definition* 9.

Say that ReactiveBayes program $\mathbf{S}$ *satisfies success condition* 1 if it decomposes as $\mathbf{S} = \mathbf{S}_1$ `and` $\mathbf{S}_2$ satisfying Condition 1, for two nonempty programs $\mathbf{S}_1$ and $\mathbf{S}_2$ themselves satisfying success condition 1. The following result holds, which relates the two different semantics, when the success condition is satisfied:

Theorem 7 (equivalence of the two semantics) *Let* $\mathbf{S}$ *be a* ReactiveBayes *program satisfying the success condition* 1*, and such that all the minimal vertices of* $[\![\mathbf{S}]\!]$ *are Mixed Systems. Then, its two semantics are probabilistically equivalent:* $[\mathbf{S}] \equiv_P [\![\mathbf{S}]\!]$.

The additional condition ensures that $\mathbf{S}$ is free from nondeterministic input variables. Thus, the minimal vertices of $[\![\mathbf{S}]\!]$ are statements of the form (48,(i),(ii)).

Proof A direct consequence of Theorem 4.

The message Passing algorithm presented in Theorem 3 allows source-to-source rewriting for mapping tree-shaped non-directed Factor Graphs to directed Mixed Bayesian Networks.

Discussion 11 (if-then-else) In Example 17, system S_1 involves an "if-then-else" statement. Syntax (46), however, does not involve such statements. This means that "if-then-else" statements are seen by syntax (46) as one instance of "f", to which no particular attention is paid. The semantics of this "f" obviously depends on the value of the Boolean control signal. However, neither the factor graph, nor the Mixed Bayesian network associated to S_1, depend on which branch is active in this "if-then-else" statement. This is harmless if the focus is on modeling. Considering "if-then-else" and paying attention to it is needed in probabilistic reasoning [20], see Appendix A.1. The same holds when performing inference or learning [37]; see also the discussion of objective 3 of probabilistic programming on page 3.

6.3 *Completing the semantics of* ReactiveBayes

Recall that the semantics of the static part of ReactiveBayes was given in (47,(i)–(v)). In this section, we first give the semantics of the dynamic fragment of ReactiveBayes (46) in terms of Mixed Automata. Then, we show that the model of Mixed Automata supports the semantics of an extension of ReactiveBayes with states and actions.

6.3.1 Semantics of the dynamic fragment of ReactiveBayes

Notations: To every variable x, we associate its successive *previous versions* ${}^{\bullet}x, {}^{\bullet 2}x, {}^{\bullet 3}x, \ldots$, where

$$ {}^{\bullet(n+1)}x =_{\text{def}} {}^{\bullet}({}^{\bullet n}x) \quad \text{and} \quad Q_{{}^{\bullet}x} = Q_x \,. \tag{50} $$

Then, we define

$$ {}^{\bullet}e(x) =_{\text{def}} e({}^{\bullet}x) \tag{51} $$

as being the expression e in which every variable x is replaced by its previous version ${}^{\bullet}x$. We will use the Mixed System $(x{=}q_x)$, defined in (34): this system has trivial probabilistic part, variable x, and enforces the value q_x for it. □

We begin with delay `pre` and initialization `init`:

$$ \begin{aligned} &(vi) && [\![\texttt{pre}\ x]\!] = \left(\{\mathrm{T}\}, \{x, {}^{\bullet}x\}, -, \{q \xrightarrow{\mathrm{T}} ({}^{\bullet}x{=}q_x) \mid \forall q \in Q\} \right) \\ &(vii) && [\![\texttt{init}\ x = c]\!] = \left(\{\mathrm{T}\}, \{x\}, c, c \xrightarrow{\mathrm{T}} \mathsf{nil} \text{ and } \epsilon \xrightarrow{\mathrm{T}} \mathsf{nil} \right) \end{aligned} \tag{52} $$

The semantics of `pre` is stated in (vi). It is the Mixed Automaton with trivial action alphabet (singleton $\{\mathrm{T}\}$), two variables x (receiving the current value) and ${}^\bullet x$ (delivering the previous value), an undefined initial state, and the set of transitions

$$q \xrightarrow{\mathrm{T}} ({}^\bullet x = q_x)\,,$$

where q ranges over the set of all states and q_x is the x-coordinate of q—this transition relation formalizes the constraint that $(\texttt{pre}\ x)_n$ holds the value of x_{n-1}.

Since the initial state is undefined in the delay statement, a specification of the initial value is required by using the initialization statement `init`. Its semantics is stated in (vii), where nil is the trivial Mixed System defined in (25). This Mixed Automaton possesses x as its only variable, $c \in Q_x$ as its initial state, and otherwise does nothing, i.e., sets no constraint on its environment.

6.3.2 Extending ReactiveBayes *with states and actions*

So far, we have completed the semantics of ReactiveBayes as defined in (46), for which actions were not used—only the trivial "true" action was used in the semantics. Since Mixed Automata is a richer framework, it can support the following richer language involving state machines by adding the following syntax, with reference to (46):

$$\begin{array}{l} \alpha ::= {}^\bullet e, \text{ where } e \text{ has Boolean type} \\ A ::= \texttt{on}\ \alpha\ \texttt{then}\ S\ \texttt{else}\ S \mid\ A\ \texttt{and}\ A \end{array} \tag{53}$$

Actions α are previous versions of expressions of Boolean type. In the additional statement "$\texttt{on}\ \alpha\ \texttt{then}\ S\ \texttt{else}\ S$", actions α and $\neg\alpha$ trigger the transition leading to the first and second system, respectively. If α is the constant "true", we simply write S instead of "$\texttt{on}$ true $\texttt{then}\ S$".

We now give the corresponding semantics (T denotes the Boolean value "true", and we refer the reader to Definition 1 regarding nil and the distinguished state ϵ):

$$(viii)\quad [\![\texttt{on}\ \alpha\ \texttt{then}\ S\ \texttt{else}\ S']\!] = \frac{S \text{ and } S' \text{ have previous state } p}{\left(\begin{array}{l} \{\alpha, \neg\alpha\}\ ,\ X \cup X'\ ,\ \cdot \\ \{p \xrightarrow{\alpha} S, p \xrightarrow{\neg\alpha} S'\} \end{array}\right)} \tag{54}$$

$$(ix)\qquad [\![A_1\ \texttt{and}\ A_2]\!] = [\![A_1]\!] \parallel [\![A_2]\!]$$

The right-hand side of $(viii)$ is an inference rule meaning "*numerator entails denominator*". By (51) and the syntax for actions in (53), action α in $(viii)$ is evaluated by using the previous state p. At a given instant, the previous state is known and can thus be used as the source state of the two transitions. The initial state is left unspecified. Focus on the parallel composition (ix). With reference to Definition 14, we now formalize the compatibility relation $\bowtie_{\mathbf{\Sigma}}$ and the join operator $\sqcup_{\mathbf{\Sigma}}$:

$$\alpha_1 \bowtie_{\mathbf{\Sigma}} \alpha_2 \text{ always holds, and } \alpha_1 \sqcup_{\mathbf{\Sigma}} \alpha_2 =_{\text{def}} \alpha_1 \wedge \alpha_2. \tag{55}$$

7 Other related work

So far, we have discussed work closely related to the different topics we covered. In this section, we broaden our discussion by considering side topics relevant to our study.

Regarding semantic studies, we did not address *denotational semantics*—our sampling (Definition 1) is an operational semantics. By denotational semantics, we mean a mathematical characterization of the set of all traces that the considered system can produce. The subject was indeed addressed in core mathematical probability theory—it was not called this way—with the Kolmogorov extension theorem: this theorem gives the denotational semantics of a sequence of independent identically μ-distributed random variables as a probability space $(\Omega, \mathcal{F}, \pi)$, where Ω is the set of trajectories, $\mathcal{F}$ the associated product σ-algebra, and $\pi = \mu^{\mathbb{N}}$, whose existence and uniqueness follows from this extension theorem. Since the 1970s, mathematicians in probability theory gave a denotational semantics (this term was not used) to stochastic differential equations in a very general setting, see e.g., the seminal paper [60]. In our context of nondeterministic/probabilistic dynamical systems, the task was not really investigated by mathematicians, and one should instead look at the literature closer to computer science. The seminal paper by Kozen [41] defines two kinds of semantics of simple imperative probabilistic programs. The first semantics has a finite horizon $[0, S]$ where S is a stopping time (causally defined random time) and closely follows probability theory with its construction of probability spaces of program traces; the second semantics, advocated by the author, is more denotational, uses Scott-like techniques of continuous linear operators on a Banach space of measures, and supports infinite traces, see also [61,34]. This approach was extended in [38,39,21] in order to provide semantics to the *observe* statement present in most modern probabilistic programming languages. In [15], the semantics of a functional language supporting mixtures of continuous and discrete distributions and dedicated to certainly terminating programs, is specified as *measure transformers*, describing how the program itself propagates the distribution of the probabilistic inputs.

Major probabilistic programming languages do offer *recursion* [19,48], all of them offer *while loops*. These features raise the issue of possible nontermination. Nonterminating while loops are the essence of [9]. We did not consider *recursion* in its full generality, but only under the limited form of nonterminating time-recursion, with Mixed Automata. Time recursion is the most widely used form of recursion considered in statistics and learning.

Inference and learning are the main concerns of probabilistic programming. Due to the generality of the considered models, Monte-Carlo based inference algorithms are preferred [37,19,32,33]. Nondeterminism, which is supported by probabilistic languages, breaks the stationarity (or time-invariance) of the specified statistical models. This is a source of difficulties when invoking limit theorems of probability theory to support learning algorithms [37]. We did not consider learning in this work. Our model of Mixed Automata would face the same challenge if inference were considered. Extension of model-based IOCO

testing with probabilities was considered in [30]—this is a different subject than statistical testing in the sense of [42].

In Section 5, we have shown that Mixed Automata subsume PA. Tutorial [58] investigates more variants of PA. We conjecture that similar results hold for these as well: mappings exist that preserve simulation but not parallel composition. Abstract Probabilistic Automata [23] are an *interface model* that support specification, not programming. In addition to parallel composition, Abstract Probabilistic Automata offer *refinement* and possess Probabilistic Automata as their *models*, two concepts irrelevant to our study.

In our work, we have considered only automata, whose dynamics is indexed by discrete time n. *Equipping true concurrency models with probability* was classical for some net models. Free choice (or confusion free) nets are models for which this is relatively simple; since choices remain local and statically defined, it is easy to turn them into probabilistic choices. However, this is no longer the case for event structures with confusion: concurrency interferes with choice, making the latter dynamically defined. This makes it intricate, to equip choices with probabilities while maximally preserving concurrency. First constructions were proposed in [4,5,6,7] based on the notion of *branching cells,* capturing the above difficulty. Infinite event structures are supported (with restrictions) for which the law of large numbers is proved. Drawbacks are: 1) different sequences of events corresponding to the same configuration may be given different probabilities, and 2) the overall probability is globally defined; hence no parallel composition can be proposed. A different construction was proposed for occurrence nets in [17,18], addressing the above drawback. The net is augmented with "negative places", thus enforcing supplementary causalities with the result of deferring choices until they become local. Through the notion of statically defined *s-cell,* the so augmented net can be given probabilistic choices meeting full concurrency, and parallel compositions of such nets is supported. In turn, the construction of the negative places works for finite nets only. In [18], a link of such augmented nets is established with Bayesian networks, thus providing a result similar to ours in Section 3. Finally, [1,2,3] study trace monoids by equipping them with probabilities derived from local specifications using analytic combinatorics techniques. As far as we know, this is the only approach supporting true concurrency with probabilistic choice and parallel composition for infinite traces. Unfortunately, concurrency makes everything more complicated.

8 Conclusion

We developed the Mixed (Probabilistic-Nondeterministic) Automata model that subsumes nondeterministic automata, probabilistic automata, and graphical probabilistic models. In a Mixed Automaton, transitions are triggered by actions and map states to Mixed Systems, from which the next state is sampled.

Mixed Systems are stateless and involve no dynamics. They combine nondeterminism and probability in a simple setting, providing an elegant theory of equivalence and a parallel composition. We proposed the notion of Mixed Kernel equipped with an incremental composition. We generalized Bayes formula by extending, to Mixed Systems and Mixed Kernels, the notions of marginal and conditional probabilities. The parallel composition of Mixed Systems naturally brings a notion of graphical structure, which subsumes Factor Graphs; similarly, the incremental composition of Mixed Kernels supports an extension of Bayesian Networks. Message-passing algorithms allow for transforming tree-shaped Factor Graphs to Bayesian Networks, as already known for the classical notions. To summarize, our model extends graphical probabilistic models to a framework in which nondeterminism and probabilities can be freely combined. This framework also subsumes Dempster's belief theory.

On top of Mixed Systems, we defined Mixed Automata and equipped them with a simulation and a parallel composition where probabilistic parts of systems can interact. This is in contrast to existing models of probabilistic automata, which do not support conditioning. Developing an interface theory with Mixed Automata as models and Abstract Probabilistic Automata [23] would make sense. We believe that the simplicity of Mixed Systems makes them an interesting candidate for the semantics of probabilistic programs—there is still a long way to go before justifying this claim.

To avoid technicalities, we decided to restrict ourselves to the consideration of finite or denumerable probability spaces. This makes the definition of support of a probability and conditional probability straightforward. Since conditioning is the heart of our approach, relaxing this restriction is far from obvious, with a deep revisiting of *consistency* for Mixed Systems. In the last appendix of [14] gives hints for such an extension.

We did not investigate decidability and complexity issues, however, neither we paid attention to effectiveness. Handling constraints C is the first difficulty. To reason on control, we could keep solving simple (e.g., Boolean) constraints, e.g., by distinguishing, in our model syntax, if-then-else statements. Other constraints may be abstracted by their associated directed or nondirected bipartite graph. Then, techniques such as the *conditional dependency graphs* of synchronous languages [12] could be adapted.

We did not investigate either the design of learning and inference algorithms, a central motivation of probabilistic programming. When considering this subject, we would encounter the problem of correct Monte-Carlo sampling in learning algorithms, which is extensively studied in [37]. In our context, this amounts to 1) identifying time-invariant model fragments, 2) applying limit theorems to them, and finally, 3) combining the results to derive learning algorithms for Mixed Systems or Automata models.

Acknowledgements: The reviewers are gratefully thanked for pointing out weaknesses and suggesting improvements, as well as providing important bibliographical items while commenting on the previous versions of this paper.

9 Conflicts of interest

The authors have no conflict of interest to declare that are relevant to this article.

References

1. Abbes, S.: Markov two-components processes. Logical Methods in Computer Science **9**(2) (2013). DOI 10.2168/LMCS-9(2:14)2013. URL https://doi.org/10.2168/LMCS-9(2:14)2013
2. Abbes, S.: Synchronization of bernoulli sequences on shared letters. Inf. Comput. **255**, 1–26 (2017). DOI 10.1016/j.ic.2017.04.002. URL https://doi.org/10.1016/j.ic.2017.04.002
3. Abbes, S.: Markovian dynamics of concurrent systems. Discrete Event Dynamic Systems **29**(4), 527–566 (2019). DOI 10.1007/s10626-019-00291-z. URL https://doi.org/10.1007/s10626-019-00291-z
4. Abbes, S., Benveniste, A.: Branching cells as local states for event structures and nets: Probabilistic applications. In: V. Sassone (ed.) Foundations of Software Science and Computational Structures, 8th International Conference, FOSSACS 2005, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2005, Edinburgh, UK, April 4-8, 2005, Proceedings, *Lecture Notes in Computer Science*, vol. 3441, pp. 95–109. Springer (2005). DOI 10.1007/978-3-540-31982-5_6. URL https://doi.org/10.1007/978-3-540-31982-5_6
5. Abbes, S., Benveniste, A.: True-concurrency probabilistic models: Branching cells and distributed probabilities for event structures. Inf. Comput. **204**(2), 231–274 (2006). DOI 10.1016/j.ic.2005.10.001. URL https://doi.org/10.1016/j.ic.2005.10.001
6. Abbes, S., Benveniste, A.: True-concurrency probabilistic models: Markov nets and a law of large numbers. Theor. Comput. Sci. **390**(2-3), 129–170 (2008). DOI 10.1016/j.tcs.2007.09.018. URL https://doi.org/10.1016/j.tcs.2007.09.018
7. Abbes, S., Benveniste, A.: Concurrency, sigma-algebras, and probabilistic fairness. In: L. de Alfaro (ed.) Foundations of Software Science and Computational Structures, 12th International Conference, FOSSACS 2009, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2009, York, UK, March 22-29, 2009. Proceedings, *Lecture Notes in Computer Science*, vol. 5504, pp. 380–394. Springer (2009). DOI 10.1007/978-3-642-00596-1_27. URL https://doi.org/10.1007/978-3-642-00596-1_27
8. Barthe, G., Espitau, T., Grégoire, B., Hsu, J., Stefanesco, L., Strub, P.: Relational reasoning via probabilistic coupling. In: M. Davis, A. Fehnker, A. McIver, A. Voronkov (eds.) Logic for Programming, Artificial Intelligence, and Reasoning - 20th International Conference, LPAR-20 2015, Suva, Fiji, November 24-28, 2015, Proceedings, *Lecture Notes in Computer Science*, vol. 9450, pp. 387–401. Springer (2015). DOI 10.1007/978-3-662-48899-7_27. URL https://doi.org/10.1007/978-3-662-48899-7_27
9. Baudart, G., Mandel, L., Atkinson, E., Sherman, B., Pouzet, M., Carbin, M.: Reactive probabilistic programming. In: Conference on Programming Language Design and Implementation (PLDI'20). London, UK (2020). To appear.
10. Bellman, R.: A markovian decision process. Journal of Mathematics and Mechanics **6**(5), 679–684 (1957). URL http://www.jstor.org/stable/24900506
11. Benveniste, A., Bourke, T., Caillaud, B., Colaço, J., Pasteur, C., Pouzet, M.: Building a hybrid systems modeler on synchronous languages principles. Proceedings of the IEEE **106**(9), 1568–1592 (2018). DOI 10.1109/JPROC.2018.2858016. URL https://doi.org/10.1109/JPROC.2018.2858016
12. Benveniste, A., Caspi, P., Edwards, S.A., Halbwachs, N., Guernic, P.L., de Simone, R.: The synchronous languages 12 years later. Proceedings of the IEEE **91**(1), 64–83 (2003)
13. Benveniste, A., Levy, B.C., Fabre, E., Guernic, P.L.: A calculus of stochastic systems for the specification, simulation, and hidden state estimation of mixed stochastic/non-stochastic systems. Theor. Comput. Sci. **152**(2), 171–217 (1995)

14. Benveniste, A., Raclet, J.: Mixed nondeterministic-probabilistic automata: Blending graphical probabilistic models with nondeterminism. CoRR **abs/2201.07474** (2022). URL https://arxiv.org/abs/2201.07474. Https://arxiv.org/abs/2201.07474
15. Borgström, J., Gordon, A.D., Greenberg, M., Margetson, J., Gael, J.V.: Measure transformer semantics for bayesian machine learning. In: ESOP, *Lecture Notes in Computer Science*, vol. 6602, pp. 77–96. Springer (2011)
16. Boudol, G.: Notes on algebraic calculi of processes. In: K.R. Apt (ed.) Logics and Models of Concurrent Systems - Conference proceedings, Colle-sur-Loup (near Nice), France, 8-19 October 1984, *NATO ASI Series*, vol. 13, pp. 261–303. Springer (1984). DOI 10.1007/978-3-642-82453-1_9. URL https://doi.org/10.1007/978-3-642-82453-1_9
17. Bruni, R., Melgratti, H.C., Montanari, U.: Concurrency and probability: Removing confusion, compositionally. Log. Methods Comput. Sci. **15**(4) (2019). DOI 10.23638/LMCS-15(4:17)2019. URL https://doi.org/10.23638/LMCS-15(4:17)2019
18. Bruni, R., Melgratti, H.C., Montanari, U.: Bayesian network semantics for petri nets. Theor. Comput. Sci. **807**, 95–113 (2020). DOI 10.1016/j.tcs.2019.07.034. URL https://doi.org/10.1016/j.tcs.2019.07.034
19. Carpenter, B., Gelman, A., Hoffman, M., Lee, D., Goodrich, B., Betancourt, M., Brubaker, M., Guo, J., Li, P., Riddell, A.: Stan: A Probabilistic Programming Language. Journal of Statistical Software, Articles **76**(1), 1–32 (2017). DOI 10.18637/jss.v076.i01. URL https://www.jstatsoft.org/v076/i01
20. Chatterjee, K., Fu, H., Novotný, P., Hasheminezhad, R.: Algorithmic analysis of qualitative and quantitative termination problems for affine probabilistic programs. ACM Trans. Program. Lang. Syst. **40**(2), 7:1–7:45 (2018). DOI 10.1145/3174800. URL https://doi.org/10.1145/3174800
21. Dahlqvist, F., Kozen, D.: Semantics of higher-order probabilistic programs with conditioning. Proc. ACM Program. Lang. **4**(POPL), 57:1–57:29 (2020). DOI 10.1145/3371125. URL https://doi.org/10.1145/3371125
22. D'Argenio, P.R., Terraf, P.S., Wolovick, N.: Bisimulations for non-deterministic labelled markov processes. Math. Struct. Comput. Sci. **22**(1), 43–68 (2012). DOI 10.1017/S0960129511000454. URL https://doi.org/10.1017/S0960129511000454
23. Delahaye, B., Katoen, J., Larsen, K.G., Legay, A., Pedersen, M.L., Sher, F., Wasowski, A.: Abstract probabilistic automata. Inf. Comput. **232**, 66–116 (2013). DOI 10.1016/j.ic.2013.10.002. URL https://doi.org/10.1016/j.ic.2013.10.002
24. Dellacherie, C., Meyer, P.: Probabilities and potentials. North-Holland Mathematics Studies, North-Holland, Amsterdam (1978)
25. Dempster, A.P.: Upper and lower probabilities induced by a multivalued mapping. Ann. Math. Statist. **38**(2), 325–339 (1967). DOI 10.1214/aoms/1177698950. URL https://doi.org/10.1214/aoms/1177698950
26. Dempster, A.P.: A generalization of bayesian inference. Journal of the Royal Statistical Society. Series B (Methodological) **30**(2), 205–247 (1968). URL http://www.jstor.org/stable/2984504
27. Doberkat, E., Terraf, P.S.: Stochastic non-determinism and effectivity functions. J. Log. Comput. **27**(1), 357–394 (2017). DOI 10.1093/logcom/exv049. URL https://doi.org/10.1093/logcom/exv049
28. Ferns, N., Panangaden, P., Precup, D.: Bisimulation metrics for continuous markov decision processes. SIAM J. Comput. **40**(6), 1662–1714 (2011). DOI 10.1137/10080484X. URL https://doi.org/10.1137/10080484X
29. Forney, D.: Maximum-likelihood sequence estimation of digital sequences in the presence of intersymbol interference. IEEE Trans. Information Theory **18**(3), 363–378 (1972). DOI 10.1109/TIT.1972.1054829. URL https://doi.org/10.1109/TIT.1972.1054829
30. Gerhold, M., Stoelinga, M.: Model-based testing of probabilistic systems. Formal Aspects Comput. **30**(1), 77–106 (2018). DOI 10.1007/s00165-017-0440-4. URL https://doi.org/10.1007/s00165-017-0440-4
31. Givan, R., Dean, T.L., Greig, M.: Equivalence notions and model minimization in markov decision processes. Artif. Intell. **147**(1-2), 163–223 (2003). DOI 10.1016/S0004-3702(02)00376-4. URL https://doi.org/10.1016/S0004-3702(02)00376-4
32. Goodman, N.D., Mansinghka, V.K., Roy, D.M., Bonawitz, K., Tenenbaum, J.B.: Church: a language for generative models. CoRR **abs/1206.3255** (2012). URL http://arxiv.org/abs/1206.3255

33. Goodman, N.D., Stuhlmüller, A.: The Design and Implementation of Probabilistic Programming Languages. http://dippl.org (2014). Accessed: 2021-4-20
34. Gordon, A.D., Henzinger, T.A., Nori, A.V., Rajamani, S.K.: Probabilistic programming. In: J.D. Herbsleb, M.B. Dwyer (eds.) Proceedings of the on Future of Software Engineering, FOSE 2014, Hyderabad, India, May 31 - June 7, 2014, pp. 167–181. ACM (2014). DOI 10.1145/2593882.2593900. URL https://doi.org/10.1145/2593882.2593900
35. Gupta, V., Jagadeesan, R., Panangaden, P.: Stochastic processes as concurrent constraint programs. In: POPL, pp. 189–202. ACM (1999)
36. Hsu, J.: Probabilistic couplings for probabilistic reasoning. CoRR **abs/1710.09951** (2017). URL http://arxiv.org/abs/1710.09951
37. Hur, C., Nori, A.V., Rajamani, S.K., Samuel, S.: A provably correct sampler for probabilistic programs. In: P. Harsha, G. Ramalingam (eds.) 35th IARCS Annual Conference on Foundation of Software Technology and Theoretical Computer Science, FSTTCS 2015, December 16-18, 2015, Bangalore, India, *LIPIcs*, vol. 45, pp. 475–488. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2015). DOI 10.4230/LIPIcs.FSTTCS.2015.475. URL https://doi.org/10.4230/LIPIcs.FSTTCS.2015.475
38. Jones, C., Plotkin, G.D.: A probabilistic powerdomain of evaluations. In: LICS, pp. 186–195. IEEE Computer Society (1989)
39. Katoen, J., Gretz, F., Jansen, N., Kaminski, B.L., Olmedo, F.: Understanding probabilistic programs. In: R. Meyer, A. Platzer, H. Wehrheim (eds.) Correct System Design - Symposium in Honor of Ernst-Rüdiger Olderog on the Occasion of His 60th Birthday, Oldenburg, Germany, September 8-9, 2015. Proceedings, *Lecture Notes in Computer Science*, vol. 9360, pp. 15–32. Springer (2015). DOI 10.1007/978-3-319-23506-6_4. URL https://doi.org/10.1007/978-3-319-23506-6_4
40. Kindermann, R., Snell, L.: Markov random fields and their applications, vol. 1. American Mathematical Society (1980). DOI http://dx.doi.org/10.1090/conm/001
41. Kozen, D.: Semantics of probabilistic programs. J. Comput. Syst. Sci. **22**(3), 328–350 (1981)
42. Lehmann, E.L., Romano, J.P.: Testing statistical hypotheses, third edn. Springer Texts in Statistics. Springer, New York (2005)
43. Loeliger, H..: An introduction to factor graphs. IEEE Signal Processing Magazine **21**(1), 28–41 (2004)
44. Lunn, D., Spiegelhalter, D., Thomas, A., Best, N.: The BUGS project: Evolution, critique and future directions. Statistics in Medicine **28**(25), 3049–3067 (2009). DOI 10.1002/sim.3680. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/sim.3680
45. Lynch, N.A., Segala, R., Vaandrager, F.W.: Compositionality for probabilistic automata. In: Proc. of the 14th International Conference on Concurreny Theory (CONCUR'03), *Lecture Notes in Computer Science*, vol. 2761, pp. 204–222. Springer (2003)
46. McIver, A., Morgan, C.: Abstraction, Refinement and Proof for Probabilistic Systems. Monographs in Computer Science. Springer (2005). DOI 10.1007/b138392. URL https://doi.org/10.1007/b138392
47. McIver, A., Morgan, C.: Correctness by construction for probabilistic programs. In: T. Margaria, B. Steffen (eds.) Leveraging Applications of Formal Methods, Verification and Validation: Verification Principles - 9th International Symposium on Leveraging Applications of Formal Methods, ISoLA 2020, Rhodes, Greece, October 20-30, 2020, Proceedings, Part I, *Lecture Notes in Computer Science*, vol. 12476, pp. 216–239. Springer (2020). DOI 10.1007/978-3-030-61362-4_12. URL https://doi.org/10.1007/978-3-030-61362-4_12
48. van de Meent, J.W., Paige, B., Yang, H., Wood, F.: An introduction to probabilistic programming (2018)
49. Olmedo, F., Gretz, F., Jansen, N., Kaminski, B.L., Katoen, J., McIver, A.: Conditioning in probabilistic programming. ACM Trans. Program. Lang. Syst. **40**(1), 4:1–4:50 (2018). DOI 10.1145/3156018. URL https://doi.org/10.1145/3156018
50. Pearl, J.: Fusion, propagation, and structuring in belief networks. Artif. Intell. **29**(3), 241–288 (1986). DOI 10.1016/0004-3702(86)90072-X. URL https://doi.org/10.1016/0004-3702(86)90072-X
51. Pearl, J.: Causality, 2 edn. Cambridge University Press, Cambridge, UK (2009). DOI 10.1017/CBO9780511803161

52. Plummer, M.: JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling. In: *Proceedings of the 3rd International Workshop on Distributed Statistical Computing (DSC 2003)* (2003). DOI 10.1007/b98484. URL https://www.r-project.org/conferences/DSC-2003/. K Hornik, F Leisch, A Zeileis (eds.)
53. Puterman, M.L.: Markov decision processes: discrete stochastic dynamic programming. John Wiley & Sons (2014)
54. Rabiner, L.R., Juang, B.H.: An introduction to hidden markov models. IEEE ASSp Magazine (1986)
55. Segala, R.: Probability and nondeterminism in operational models of concurrency. In: Proc. of the 17th International Conference on Concurrency Theory (CONCUR'06), *Lecture Notes in Computer Science*, vol. 4137, pp. 64–78. Springer (2006)
56. Segala, R., Lynch, N.A.: Probabilistic simulations for probabilistic processes. In: Proc. of the 5th International Conference on Concurrency Theory (CONCUR'94), *Lecture Notes in Computer Science*, vol. 836, pp. 481–496. Springer (1994)
57. Shafer, G.: A Mathematical Theory of Evidence. Princeton University Press, Princeton (1976)
58. Sokolova, A., de Vink, E.P.: Probabilistic automata: System types, parallel composition and comparison. In: C. Baier, B.R. Haverkort, H. Hermanns, J. Katoen, M. Siegle (eds.) Validation of Stochastic Systems - A Guide to Current Research, *Lecture Notes in Computer Science*, vol. 2925, pp. 1–43. Springer (2004). DOI 10.1007/978-3-540-24611-4_1. URL https://doi.org/10.1007/978-3-540-24611-4_1
59. Staton, S., Yang, H., Wood, F.D., Heunen, C., Kammar, O.: Semantics for probabilistic programming: higher-order functions, continuous distributions, and soft constraints. In: M. Grohe, E. Koskinen, N. Shankar (eds.) Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '16, New York, NY, USA, July 5-8, 2016, pp. 525–534. ACM (2016). DOI 10.1145/2933575.2935313. URL https://doi.org/10.1145/2933575.2935313
60. Stroock, D.W., Varadhan, S.R.S.: On the support of diffusion processes with applications to the strong maximum principle. In: Proceedings of the Sixth Berkeley Symposium on Mathematical Statistics and Probability, Volume 3: Probability Theory, pp. 333–359. University of California Press, Berkeley, Calif. (1972). URL https://projecteuclid.org/euclid.bsmsp/1200514345
61. Tix, R., Keimel, K., Plotkin, G.: Semantic domains for combining probability and nondeterminism. Electr. Notes Theor. Comput. Sci. **222**, 3–99 (2009)
62. Tolpin, D., Zhou, Y., Rainforth, T., Yang, H.: Probabilistic programs with stochastic conditioning. In: M. Meila, T. Zhang (eds.) Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event, *Proceedings of Machine Learning Research*, vol. 139, pp. 10312–10323. PMLR (2021). URL http://proceedings.mlr.press/v139/tolpin21a.html
63. Wang, D., Hoffmann, J., Reps, T.W.: A denotational semantics for low-level probabilistic programs with nondeterminism. In: B. König (ed.) Proceedings of the Thirty-Fifth Conference on the Mathematical Foundations of Programming Semantics, MFPS 2019, London, UK, June 4-7, 2019, *Electronic Notes in Theoretical Computer Science*, vol. 347, pp. 303–324. Elsevier (2019). DOI 10.1016/j.entcs.2019.09.016. URL https://doi.org/10.1016/j.entcs.2019.09.016

In this supplementary material, we further detail the comparison between our model and imperative probabilistic programming. Then, we collect all the missing proofs.

A Addendum and Proofs Regarding Mixed Systems

A.1 Comparison with imperative probabilistic programming, see Discussion 1

In this appendix, we compare our model of Mixed Systems with imperative probabilistic programming following the approach promoted by Mc Iver and Morgan [46,47]. This line of work addresses probabilistic extensions of Hoare logic for imperative programs, focusing on evaluating the probability of the weakest preconditions of properties. In addition, we like to compare our approach with one aspect of this work, namely the modeling of the blending of probability and nondeterminism—this is only a minor aspect of the work of Mc Iver and Morgan, which focuses on decidability issues and computational cost of their proposed logic.

A.1.1 Demonic/angelic nondeterminism

We choose to base our comparison on a different work in the same direction: [20], which provides the most extensive development on *demonic/angelic* blending of probability and nondeterminism in the language Apps. We do not claim to cover all aspects of Apps, since this reference focuses on checking almost sure termination using supermartingale techniques. Since our scope is more modest in this appendix, we will only develop an informal comparison based on the following example corresponding to Fig. 2 of [20], reproduced here as Figs. 6 and 7.

The program and its semantics are self-speaking. A key point here is the role of demonic and angelic nondeterminisms, and their combination in this program. Let us consider the post-condition

$$P : x \text{ gets increased by one by performing } Q_3\,. \tag{56}$$

The question is: how do we assess P? Under demonic choice, P is violated if there exists some branch in the nondeterministic choice under which P is violated. Under angelic choice, P is violated if, for all branches in the nondeterministic choice, P is violated. Inspecting Fig. 7 shows that P is violated if and only if Q_1 is selected. Thus the probabilistic score that P is violated is 0.6—we do not use the term "probability" since P combines both probabilistic and nondeterministic features and cannot be given a true probability.

Can we cast this example into Mixed Systems?

A.1.2 Casting this example to Mixed Systems?

Consider the following attempt by defining the Mixed System $S_{Q_3} = \{(\Omega, \pi), C, \{x, x'\}\}$, where:

- $\Omega = \{Q_1, Q_2\}$ and $\pi(\omega{=}Q_1) = 0.6, \pi(\omega{=}Q_2) = 0.4$;
- Variable x, x' correspond to the statuses of variable x of Q_3 from Fig. 7, before and after executing Q_3; the value of x is assumed and the value of x' will be established by sampling S_{Q_3};
- It remains to define relation C involving ω, x, x'. To mimic Fig. 7, we would like to write something like

$$x' = \textbf{if } \omega = Q_1 \textbf{ then angel } x' \in \{x-1, x+1\} \\ \textbf{else demon } x' \in \{x-1, x+1\}$$

Unfortunately, **angelic/demonic choices** are not concepts of our Mixed Systems model following Definition 1. Concerning probabilistic evaluation of state properties (item 3 of Definition 1), we could specify whether we use $\overline{\pi}$ (mirroring demonic) or $\underline{\pi}$ (mirroring angelic). Still, this does not allow to combine of both alternatives for different parts of the system.

```
x := 0;
while x ≥ 0 do
  if prob(0.6) then
    if angel then
      x := x + 1
    else
      x := x − 1
    fi                    } Q1
  else
    if demon then
      x := x + 1
    else
      x := x − 1
    fi                    } Q2
  fi
od
```

Verbatim from [20]: There is only one program variable x and no random variables. There is a while loop, where given a probabilistic choice, one of two statement blocks Q_1 or Q_2 is executed. The block Q_1 (resp., Q_2) is chosen to execute stochastically w.r.t. the probabilistic choice (Q_1 *is selected with probability* 0.6). The statement block Q_1 (resp., Q_2) is an angelic (resp., demonic) conditional statement to either increment or decrement x.
Following [20], call Q_3 the body of the `while` loop of this example: `while` $x \geq 0$ `do` Q_3.

Fig. 6 Example of Fig. 2 of [20]

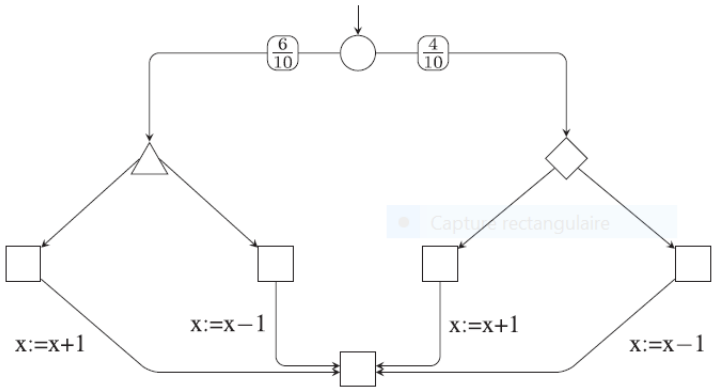


Fig. 7 Semantics: SGS (Stochastic Game Structure) of Q_3, Fig. 6 of [20]. The execution begins with the probabilistic choice. The left branch (corresponding to Q_1) is selected according to demonic nondeterminism figured by a triangle, and the right branch (corresponding to Q_2) is selected according to angelic nondeterminism, figured by a diamond.

We propose to refine Definition 1 so that both types of nondeterminism can be freely combined. Let us investigate this in the above example. Consider the Mixed System

$$S = (\Omega, \pi, X, C)\,, \tag{57}$$

where:

- $\Omega = \{Q_1, Q_2\}$ and $\pi(\omega = Q_1) = 0.6, \pi(\omega = Q_2) = 0.4$;
- Variable x, x' correspond to the statuses of variable x of Q_3 from Fig. 7, before and after executing Q_3; the value of x is assumed and the value of x' will be established by sampling S_{Q_3};
- Relation C is (yet informally) defined by

$$\omega C x' \text{ iff } \begin{cases} \omega = Q_1 \wedge \quad \textbf{angel } x' \in \{x{-}1, x{+}1\} \\ \text{or} \\ \omega = Q_2 \wedge \textbf{demon } x' \in \{x{-}1, x{+}1\} \end{cases} \tag{58}$$

This definition for C is informal, since keywords **demon** and **angel** have no mathematical meaning by themselves. We will give a semantics to (58) by assigning, to each state predicate,

a *probabilistic score* π^*. More precisely, we define $\pi^*(\neg P)$, the probabilistic score of predicate $\neg P$, by the following formula:

$$\begin{array}{rl} \pi^*(\neg P) & =_{\text{def}} \quad \pi^{\mathsf{c}}\left(\{\omega \mid \omega{=}Q_1 \wedge \exists x'{\in}\{x{-}1, x{+}1\} : \neg P\}\right) \\ & \quad +\pi^{\mathsf{c}}\left(\{\omega \mid \omega{=}Q_2 \wedge \forall x'{\in}\{x{-}1, x{+}1\} : \neg P\}\right) \end{array} \tag{59}$$

In this formula, we give a semantics to **angel** in (58) by using the existential quantifier, i.e., we use the outer probability to evaluate the corresponding state predicate; we give a semantics to the **demon** in (58) by using the universal quantifier, i.e., we use the inner probability to evaluate the corresponding state predicate. Now, for this example, $\pi^{\mathsf{c}} = \pi$ since, with relation (58), for both choices $\omega = Q_1$ and $\omega = Q_2$, corresponding values for state x' exist. Formula (59) finally yields $\pi^*(\neg P) = 0.6$.

The above coding applies only to a restricted class of relations C. In formula (59), we exploited the fact that, in relation C defined by (58), a partition of Ω is performed first (probabilistic choice). Then, each branch of this choice involves a pure state predicate independent from ω.

Here are some hints to extend this link beyond the particular example. Our starting point is the semantics of Apps, which is expressed in terms of *Stochastic Game Structures* (SGS); see Definition 2.3 of [20]. Since Mixed Systems do not support recursion, we consider only the subclass of SGS that are DAGs. Then, picking a probabilistic location ℓ of this SGS, we consider the maximal subgraph of this SGS that has ℓ as its only minimal location, and contains no other probabilistic location. For our example (57,58,59), this yields the whole SGS. For each such subgraph, a coding similar to (57,58,59) can be given. The partially ordered execution of the whole SGS is then mapped to a Bayesian network following Definition 9. The incremental sampling of this Bayesian Network would correspond to the execution of the SGS as a game.

We preferred not to refine our Mixed System model with this additional feature since: 1) it applies only to a restricted class of relations C, and 2) we believe it to be incompatible with having a parallel composition.

A.2 Proof of Lemma 2

Proof It is enough to prove the result for compressed systems. For $i = 1, 2$, let $S_i \equiv S'_i$ and let φ_i be the bijections defining the two equivalences. We define

$$\varphi(\omega, q_1 \sqcup q_2) = \big((\omega'_1, \omega'_2), q'_1 \sqcup q'_2\big) \text{ where } (\omega'_i, q'_i) = \varphi_i(\omega_i, q_i), i = 1, 2$$

and we have to verify that φ defines the desired equivalence between $S =_{\text{def}} S_1 \parallel S_2$ and $S' =_{\text{def}} S'_1 \parallel S'_2$. Using the fact that $\pi = \pi_1 \times \pi_2$, we get

$$\begin{array}{rl} C_\pi & = \{(p_1 \sqcup p_2, \omega, q_1 \sqcup q_2) \mid q_1 \bowtie q_2 \ \wedge\ \omega_1 C_1 q_1 \ \wedge\ \pi_1(\omega_1){>}0 \ \wedge\ \omega_2 C_2 q_2 \ \wedge\ \pi_2(\omega_2){>}0\} \\ & = \{(p, \omega, q_1 \sqcup q_2) \mid q_1 \bowtie q_2 \ \wedge\ \omega_1 C_{1\pi} q_1 \ \wedge\ \omega_2 C_{2\pi} q_2\} \end{array}$$

Thus, for every $(p, \omega, q_1 \sqcup q_2) \in C_\pi$, we have $q'_1 = q_1 \bowtie q_2 = q'_2$ and $\omega'_i C_{i\pi} q'_i, i = 1, 2$, whence $\omega' C'_\pi q'$ and φ is a bijection. Since $\pi' = \pi'_1 \times \pi'_2$ we get $\pi'(\omega') = \pi(\omega)$, which finishes the proof.

B Proofs Regarding Mixed Bayesian Networks

B.1 Proof of Theorem 2

Proof We will repeatedly use notation (34). Without loss of generality we can assume that S is compressed. We first compress $\mathbf{Margin}_Y(S)$ by considering the following equivalence

relation, where $Z = X \setminus Y$ and q_Y, q_Z are valuations for Y and Z:

$$\omega' \sim_Y \omega \text{ iff } \forall q_Y : \begin{cases} \exists q_Z : \omega C(q_Y, q_Z) \\ \Updownarrow \\ \exists q'_Z : \omega' C(q_Y, q'_Z) \end{cases} ; \text{ let } \omega_Y \text{ be the equivalence class of } \omega.$$

Let

$$C_Y =_{\text{def}} \{(\omega_Y, q_Y) \in \Omega_Y \times Q_Y \mid \exists \omega \in \omega_Y : \omega \, \mathbf{Pr}_Y(C) \, q_Y\}$$

be the associated relation, and let π_Y be the compressed probability defined by $\pi_Y(\omega_Y) = \sum_{\omega \in \omega_Y} \pi(\omega)$. Let us denote by

$$S_Y = (\Omega_Y, \pi_Y, Y, C_Y)$$

the resulting compressed system, and we recall that $\Omega_Y^{\mathsf{c}} = \{\omega_Y \mid \exists q_Y : \omega_Y C_Y q_Y\}$. In the sequel, we feel free to identify $\omega_Y \in \Omega_Y$, an element of the set of equivalence classes, with ω_Y seen as a subset of Ω saturated for $\sim_Y$. This way, a subset of Ω_Y can also be interpreted as a subset of Ω.

To prove the theorem, we compare the two probabilistic semantics, namely: which state can be output and what is the outer probability of producing it. By definition of the sequential composition of kernels, $\mathbf{Margin}_Y(S) \,;\, \mathbf{Cond}_Y(S)$

1. samples $\mathbf{Margin}_Y(S) \rightsquigarrow q_Y$; and, then
2. given q_Y, samples $(Y{=}q_Y) \parallel S$.

Regarding the relations governing the nondeterministic choice, the combination of these two steps is identical to C. Let q_* be such that $S \rightsquigarrow q_*$, implying that $\mathbf{Margin}_Y(S) \rightsquigarrow q_{*Y}$, where $q_{*Y} =_{\text{def}} \mathbf{Pr}_Y(q_*)$. Let us evaluate the outer probabilistic score of q_* for the Bayesian network $\mathbf{Margin}_Y(S) \,;\, \mathbf{Cond}_Y(S)$, i.e., the probability that q_* is a possible outcome of sampling $\mathbf{Margin}_Y(S) \,;\, \mathbf{Cond}_Y(S)$. We need to prove that it is equal to the probability that q_* is a possible outcome of S, namely $\pi^{\mathsf{c}}(C_{q_*})$—we used notation (29). To show this, we note the following:

1. To output q_* we first must output q_{*Y}, which amounts to selecting ω_Y such that $\omega_Y C_Y q_{*Y}$. Using (12), (28) and notation (29), the probabilistic score of q_{*Y}, i.e., the probability that q_{*Y} is a possible outcome of $\mathbf{Margin}_Y(S)$, is equal to
$$\pi_Y^{\mathsf{c}}\big((C_Y)_{q_{*Y}}\big) \tag{60}$$
which is > 0 since $\mathbf{Margin}_Y(S) \rightsquigarrow q_{*Y}$.
2. Then, we must select ω using S, under the additional constraint that $\mathbf{Pr}_Y(q) = q_{*Y}$, which requires that we sample $\omega \in \Omega$ under the constraint that $\omega \in \omega_Y$ for some $\omega_Y \in (C_Y)_{q_{*Y}}$. The corresponding probabilistic score is thus equal to the conditional probability
$$\pi^{\mathsf{c}}\big(C_{q_*} \mid (C_Y)_{q_{*Y}}\big)\,, \tag{61}$$
which is well defined since $\pi_Y^{\mathsf{c}}((C_Y)_{q_{*Y}}) > 0$.
3. By (33), the probabilistic score of q_* is equal to the product of the two scores (60) and (61):
$$\pi^{\mathsf{c}}\big(C_{q_*} \mid (C_Y)_{q_{*Y}}\big)\, \pi_Y^{\mathsf{c}}\big((C_Y)_{q_{*Y}}\big) \;=\; \pi^{\mathsf{c}}\big(C_{q_*} \cap (C_Y)_{q_{*Y}}\big) \;=\; \pi^{\mathsf{c}}\big(C_{q_*}\big)\,,$$
where the last equality follows from $C_{q_*} \subseteq (C_Y)_{q_{*Y}}$.

This shows that q_* possesses identical probabilistic semantics, for the left and right hand side of Bayes formula.

B.2 Proof of Corollary 1

As a prerequisite, we need the following result:

Lemma 5 *Let S_1 and S_2 be any two Mixed Systems, and let Y be a set of variables containing $X_1 \cap X_2$. Then, we have:* $\mathbf{Margin}_{X_1 \cup Y}(S_1 \parallel S_2) \equiv S_1 \parallel \mathbf{Margin}_Y(S_2)$.

Proof This is immediate by observing that, first, $\mathbf{Margin}_{X_1 \cup Y}(S_1 \parallel S_2)$ on the one hand, and $S_1 \parallel \mathbf{Margin}_Y(S_2)$ on the other hand, possess identical probability spaces, namely $(\Omega_1, \pi_1) \times (\Omega_2, \pi_2)$, and, second, they possess identical relations $\mathbf{Pr}_{X_1 \cup Y}(C_1 \wedge C_2) = C_1 \wedge \mathbf{Pr}_{X_1 \cup Y}(C_2) = C_1 \wedge \mathbf{Pr}_Y(C_2)$. □

With this lemma, we can now proceed to the proof of Corollary 1. For proving formula (35), we first apply Theorem 2 with S replaced by $S_1 \parallel S_2$, which yields: $S_1 \parallel S_2 \equiv_P \mathbf{Margin}_{X_1 \cup Y}(S_1 \parallel S_2) ; \mathbf{Cond}_Y(S_1 \parallel S_2)$. Then, by Lemma 5, $\mathbf{Margin}_{X_1 \cup Y}(S_1 \parallel S_2) \equiv S_1 \parallel \mathbf{Margin}_Y(S_2)$ and then we conclude by observing that

$$\big(S_1 \parallel \mathbf{Margin}_Y(S_2)\big) ; \mathbf{Cond}_Y(S_1 \parallel S_2) \;\equiv_P\; \big(S_1 \parallel \mathbf{Margin}_Y(S_2)\big) ; \mathbf{Cond}_Y(S_2) \;,$$

since the outcome of S_1 is determined by the left hand factor of ";".

B.3 Proof of Theorem 3

Having proved Lemma, the proof of Theorem 3 reproduces exactly the reasoning steps establishing the message-passing algorithm mapping factor graphs to Mixed Bayesian Networks in the classical setting [43]; thus, we only sketch the argument of the proof here.

Proof Since $\mathcal{G}_\mathcal{S}$ is a tree, a natural distance can be defined on the set of vertices of $\mathcal{G}_\mathcal{S}$ by taking the length of the unique path linking two vertices. Select an arbitrary system S_o as an origin and partially order other systems according to their distance to the origin, let $\preceq$ be this partial order. We have thus made $\mathcal{G}_\mathcal{S}$ a rooted tree, which we can see as a DAG. Then, the following two rules, known as *message passing,* are considered:

R1: Pick $S \in \mathcal{G}_\mathcal{S}$, let $S^\uparrow$ be its (unique) ancestor in the tree and let $X^\uparrow$ be the set of common variables of $S^\uparrow$ and S. Then, let $S^<$ denote the parallel composition of all strict ancestors of S in $\mathcal{G}_\mathcal{S}$ and let $X^<$ be the set of variables of $S^<$. Using Bayes formula, factor S as

$$S \;\equiv_P\; \mathbf{Margin}_{X^\uparrow}(S) ; \mathbf{Cond}_{X^\uparrow}(S) \;\equiv_P\; \mathbf{Margin}_{X^<}(S) ; \mathbf{Cond}_{X^<}(S) \;,$$

where the second equivalence follows from the fact that additional variables belonging to $X^< \setminus X^\uparrow$ are not shared with S.

R2: Using formula (35) of Lemma 1, reorganize $\mathcal{S}$ by rewriting

$$S^< \parallel S \;\equiv_P\; \big(S^< \parallel \mathbf{Margin}_{X^<}(S)\big) \;;\; \mathbf{Cond}_{X^<}(S) \;.$$

Rules R1 followed by R2 are successively applied starting from the leaves of the tree, down to its root. The result is a Mixed Bayesian Network.

B.4 Proof of Theorem 4

The proof is by induction over the cardinality n of the set $\mathbb{K}$ of Mixed Kernels involved in $\mathcal{N}$. Its induction step uses the formula (36) of Corollary 2.

If $n = 1$, then, by Condition 1, $\mathcal{N}$ is a Mixed Kernel K with no input, i.e., a Mixed System S by Convention 1. By (37), we have $S \equiv S_K$, hence $S_K \equiv_P K$. Thus, $\mathcal{N}$ and $S_\mathcal{N}$ are probabilistically equivalent.

We assume the theorem is true for a cardinality of $n-1$ and prove it for a cardinality of n. Chose $K_* \in \mathbb{K}$ such that K is maximal for the order defined by DAG $\mathcal{N}$, and let $\mathcal{N}_1$ be the restriction of $\mathcal{N}$ to $\mathbb{K} - \{K_*\}$ and $\mathcal{N}_2 =_{\text{def}} \{K_*\}$. Then, $\mathcal{N} = \mathcal{N}_1 \cup \mathcal{N}_2$ by construction. Referring to (38), set $S =_{\text{def}} S_{\mathcal{N}}, S_1 =_{\text{def}} S_{\mathcal{N}_1}$, and $S_2 =_{\text{def}} S_{\mathcal{N}_2}$.

First, $S \equiv S_1 \parallel S_2$ holds by construction of the embedding $\mathcal{N} \mapsto S_{\mathcal{N}}$ defined in (38). Next, since the inputs of $\mathcal{N}_2$ are outputs of $\mathcal{N}_1$ and by Condition 2 of Theorem 4, it follows that

$$S_1 \parallel \mathbf{Margin}_Y(S_2) \equiv S_1 \,,$$

where Y is the set of shared variables between S_1 and S_2. Hence, we can apply formula (36) of Corollary 2 to the pair (S_1, S_2) of Mixed Systems, which yields:

$$S \equiv S_1 \parallel S_2 \equiv_P S_1; \mathbf{Cond}_Y(S_2) \,. \tag{62}$$

By induction hypothesis, we have

$$S_1 \equiv_P \mathcal{N}_1 \text{ and } S_2 \equiv_P \mathcal{N}_2 \,. \tag{63}$$

By definition of the binary operator ";", see (32), the following identity holds:

$$\mathcal{N}_1; \mathbf{Cond}_Y(S_2) \;=\; \mathcal{N}_1 \cup \mathcal{N}_2 \,.$$

Combining this identity with (62) and (63) finishes the proof of the induction step.

C Proofs Regarding Mixed Automata

C.1 Proof of Lemma 3

Proof The result is immediate if both S_1 and S_1' are compressed, see Definition 4. It is thus sufficient to prove the lemma for the following two particular cases: S_1 compresses to S_1', and the converse.

Consider first the case: S_1 compresses to S_1'. Let $w(\omega_1, \omega_2)$ be the weighting function associated to the lifting $S_1 \, \rho^{\mathbb{S}} \, S_2$, and let $\pi_1'(\omega_1') = \sum_{\omega_1 \in \omega_1'} \pi_1(\omega_1)$ be the relation between π_1' and π_1 in the compression of S_1 to S_1'. Then $w'(\omega_1', \omega_2) = \sum_{\omega_1 \in \omega_1'} w(\omega_1, \omega_2)$ defines the weighting function associated to the lifting $S_1' \, \rho^{\mathbb{S}} \, S_2$. The other properties required to deduce $S_1' \, \rho^{\mathbb{S}} \, S_2$ are immediate to prove.

Now, consider the alternative case: S_1' compresses to S_1, with relation

$$\pi_1(\omega_1) = \textstyle\sum_{\omega_1' \in \omega_1} \pi_1'(\omega_1') \tag{64}$$

between π_1' and π_1, where $\omega_1' \in \omega_1$ means that ω_1 is the equivalence class of ω_1' with respect to relation $\sim$ defined in (19) when compressing S_1'. This case is slightly more involved since the weighting function $w'(\omega_1', \omega_2)$ needs to be constructed. We need $w'(\omega_1', \omega_2)$ to satisfy the following relations:

$$\begin{array}{c} \forall \omega_1' : \pi_1'(\omega_1') = \sum_{\omega_2} w'(\omega_1', \omega_2) \\ \forall \omega_2 : \pi_2(\omega_2) = \sum_{\omega_1'} w'(\omega_1', \omega_2) \\ \forall (\omega_1', \omega_2; q_1) : \begin{bmatrix} w'(\omega_1', \omega_2) > 0 \\ \omega_1' \; C_1' \; q_1 \end{bmatrix} \Rightarrow \exists q_2 : \begin{bmatrix} \omega_2 \; C_2 \; q_2 \\ q_1 \; \rho \; q_2 \end{bmatrix} \end{array} \tag{65}$$

Focus first on the first two lines of (65). The following calculation shows that

$$w'(\omega_1', \omega_2) \;=_{\text{def}}\; w(\omega_1, \omega_2) \times \frac{\pi_1'(\omega_1')}{\pi_1(\omega_1)} \times \mathbf{1}(\pi_1(\omega_1) > 0) \,,$$

where ω_1 is such that $\omega'_1 \in \omega_1$ and $\mathbf{1}(B)$ equals 1 if predicate B is true and 0 otherwise, yields a weighting function w' satisfying the first two lines of (65):

$$\begin{aligned}
\sum_{\omega_2} w'(\omega'_1,\omega_2) &= \sum_{\omega_2}\left(w(\omega_1,\omega_2)\times\frac{\pi'_1(\omega'_1)}{\pi_1(\omega_1)}\times\mathbf{1}(\pi_1(\omega_1){>}0)\right)\\
&= \frac{\pi'_1(\omega'_1)}{\pi_1(\omega_1)}\times\mathbf{1}(\pi_1(\omega_1){>}0)\times\sum_{\omega_2} w(\omega_1,\omega_2) \;=\; \pi'_1(\omega'_1)\\
\sum_{\omega'_1} w'(\omega'_1,\omega_2) &= \sum_{\omega'_1}\left(w(\omega_1,\omega_2)\times\frac{\pi'_1(\omega'_1)}{\pi_1(\omega_1)}\times\mathbf{1}(\pi_1(\omega_1){>}0)\right)\\
&= \sum_{\omega_1}\left(w(\omega_1,\omega_2)\times\frac{1}{\pi_1(\omega_1)}\times\mathbf{1}(\pi_1(\omega_1){>}0)\right)\underbrace{\sum_{\omega'_1\in\omega_1}\pi'_1(\omega'_1)}_{=\pi_1(\omega_1)}\\
&= \sum_{\omega_1} w(\omega_1,\omega_2) \;=\; \pi_2(\omega_2)\,.
\end{aligned}$$

We move to the third line of (65). The conditions $w'(\omega'_1,\omega_2) > 0$ and $\omega'_1\, C'_1\, q_1$ together imply $w(\omega_1,\omega_2) > 0$ and $\omega_1\, C_1\, q_1$ where ω_1 is the equivalence class of ω'_1, i.e., $\omega'_1 \in \omega_1$. The right hand side then follows since we have $S_1\ \rho^{\mathbb{S}}\ S_2$. This finishes the proof.

C.2 Proof of Lemma 4

Proof Set $M' =_{\text{def}} M'_1 \parallel M'_2$ and $M =_{\text{def}} M_1 \parallel M_2$. Define the relation $\leq$ between Q' and Q by: $q' \leq q$ iff $q'_1 \leq_1 q_1$ and $q'_2 \leq_2 q_2$. Let us prove that $\leq$ is a simulation. Let q' be such that $q' \xrightarrow{\alpha}_{M'} S'$ for some consistent S'. Then, $q' = q'_1 \sqcup q'_2$ and $S' = S'_1 \parallel S'_2$. By definition of the parallel composition, we have $q'_i \xrightarrow{\alpha_i}_{M'_i} S'_i$ for $i = 1,2$, with $\alpha_1 \bowtie_{\mathbf{\Sigma}} \alpha_2$ and $\alpha = \alpha_1 \sqcup_{\mathbf{\Sigma}} \alpha_2$. Since $q'_i \leq q_i$, we derive the existence (and uniqueness) of consistent systems $S_i, i = 1,2$ such that $q_i \xrightarrow{\alpha_i}_{M_i} S_i$. Since $q = q_1 \sqcup q_2$ we have $q_1 \bowtie q_2$ and, thus, by definition of the parallel composition, we deduce $r \xrightarrow{\alpha}_M S_1 \parallel S_2$. It remains to show that $S_1 \parallel S_2$ is consistent. To prove this, remember that $S' = S'_1 \parallel S'_2$ is consistent. Thus, there exist compatible q'_1 and q'_2 such that $S'_i \rightsquigarrow q'_i, i = 1,2$. By definition of the simulations $\leq_i$, we deduce that $S_i \rightsquigarrow q_i, i = 1,2$, which shows that $S_1 \parallel S_2$ is consistent.

D Proofs Regarding the comparison with Probabilistic Automata

D.1 Proof of Theorem 5 regarding Simple Probabilistic Automata

D.1.1 Statement 1 of Theorem 5: from SPA to Mixed Automata

Proof The sampling of SPA P is: if P is in state $q{\in}Q$, performing $\alpha{\in}\Sigma$ leads to some target set of probability distributions over Q, of which one is selected, nondeterministically, and used to draw at random the next state q'.

We can reinterpret this sampling as follows: performing $\alpha{\in}\Sigma$ while being in state $q{\in}Q$ leads to the same target set of probability distributions over Q, that we use differently. We form the direct product of all distributions belonging to the target set and, we perform one trial according to this distribution, i.e., we perform independent random trials for all probabilities belonging to the target set. This yields a tuple of candidate values for the next state, of which we select one nondeterministically.

Clearly, these two samplings produce identical outcomes. The latter is the sampling of Mixed Automaton

$$M_P = (\Sigma, \{\xi\}, q_0, \rightarrow_P), \tag{66}$$

defined as follows:

1. Alphabet Σ of M_P is identical to that of P;
2. The unique variable ξ of M_P enumerates the values of Q, and initial state q_0 is identical to that of P; hence, P and M_P possess identical sets of states, related via the identity map;
3. $\rightarrow_P$ maps a pair $(q, \alpha) \in Q \times \Sigma$ to the mixed system $S(q) = (\Omega, \Pi, \xi, q, C)$, where:
   (a) Ω is the product of n copies of Q, where n is the cardinality of the set $\{\pi \mid (q, \alpha, \pi) \in \rightarrow\}$; thus, ω is an n-tuple of states: $\omega = (q_1, \ldots, q_n)$.
   (b) Π is the product of all probabilities belonging to $\{\pi \mid (q, \alpha, \pi) \in \rightarrow\}$;
   (c) Relation C is defined by $(\omega, q') \in C$ if and only if $q' \in \{q_1, \ldots, q_n\}$.

So, we map SPA P to Mixed Automaton M_P, defined in (66).

Mapping simulation relations: Defining simulation relations for PA requires lifting relations from states to distributions over states. The formal definition for this lifting, as given in Section 4.1 of [55], corresponds to our Definition 15 when restricted to purely probabilistic mixed systems. The same holds for the strong simulation relation defined in Section 4.2 of the same reference: it is verbatim our Definition 16 when restricted to purely probabilistic mixed systems. This proves the part of Theorem 5 regarding simulation.

Mapping parallel composition: We move to parallel composition, for which the reader is referred to [45], Section 3. For $P_1 = (\Sigma, Q_1, q_{0,1}, \rightarrow_1)$ and $P_2 = (\Sigma, Q_2, q_{0,2}, \rightarrow_2)$ two PA, their parallel composition is $P = P_1 \parallel P_2 = (\Sigma, Q_1 \times Q_2, (q_{0,1}, q_{0,2}), \rightarrow)$, where

$$(q_1, q_2) \xrightarrow{\alpha} \pi_1 \times \pi_2 \quad \text{iff} \quad q_i \xrightarrow{\alpha}_i \pi_i \text{ for } i = 1, 2 \tag{67}$$

So, on one hand we consider the Mixed Automaton M_P. On the other hand, we consider the parallel composition of their images M_{P_1} and M_{P_2}, namely $M = M_{P_1} \parallel M_{P_2} = (\Sigma, \{\xi_1, \xi_2\}, (q_{0,1}, q_{0,2}), \rightarrow_{12})$. In M, the state space is the domain of the pair (ξ_1, ξ_2), namely $Q_1 \times Q_2$, and, since there is no shared variable between the two Mixed Automata, the transition relation $\rightarrow_{12}$ is given by:

$$(q_1, q_2) \xrightarrow{\alpha}_{12} S_1 \parallel S_2 \quad \text{iff} \quad q_i \xrightarrow{\alpha}_i S_i \text{ for } i = 1, 2 \tag{68}$$

We thus need to show that

$$M_P \text{ and } M \text{ are simulation equivalent.} \tag{69}$$

We will actually show that the identity relation between the two state spaces (both are equal to $Q_1 \times Q_2$) is a simulation relation in both directions.

Observe first that (67) and (68) differ in that the former involves a nondeterministic transition relation, whereas the latter involves a deterministic transition function, mapping states to mixed systems. Pick $(q_1, q_2) \in Q_1 \times Q_2$ and consider a transition for M_P:

$$(q_1, q_2) \xrightarrow{\alpha}_{M_P} S = ((\Omega, \Pi), \xi, (q_1, q_2), C)$$

where we have, for S:

- Ω is the product of n_1 copies of Q_1 and n_2 copies of Q_2, where, for $i = 1, 2$, n_i is the cardinality of the set $\{\pi_i \mid (q_i, \alpha, \pi_i) \in \rightarrow_i\}$, so that ω identifies $n_1 \times n_2$-tuple of states: $\omega = (q_{11}, \ldots, q_{1n_1}; q_{21}, \ldots, q_{2n_2})$;
- Π is the product of all probabilities belonging to set $\{\pi_1 \times \pi_2 \mid (q_i, \alpha, \pi_i) \in \rightarrow_i\}$;
- ξ has domain $Q_1 \times Q_2$;
- $(\omega, (q_1'', q_2'')) \in C$ if and only if

$$(q_1'', q_2'') \in \{(q_{1i_1}, q_{2i_2}) \mid i_1 \in \{1, \ldots, n_1\} \text{ and } i_2 \in \{1, \ldots, n_2\}\}.$$

Next, pick $(q_1, q_2) \in Q_1 \times Q_2$ and consider a transition for M, see (68). We need to detail what $S_1 \parallel S_2 = ((\Omega', \Pi'), \xi', (q_1, q_2), C')$ is. We have, for $S_1 \parallel S_2$:

- Ω' is still the product of n_1 copies of Q_1 and n_2 copies of Q_2;
- Π' is the product $\Pi_1 \times \Pi_2$, where Π_i is the product of all probabilities belonging to set $\{\pi_i \mid (q_i, \alpha, \pi_i) \in \rightarrow_i\}$;
- ξ' has domain $Q_1 \times Q_2$;
- $(\omega, (q_1', q_2')) \in C'$ if and only if

$$(q_1', q_2') \in \{(q_{1i_1}, q_{2i_2}) \mid i_1 \in \{1, \ldots, n_1\} \text{ and } i_2 \in \{1, \ldots, n_2\}\}\,.$$

By associativity of $\times$, $\Pi' = \Pi$, whereas other items for S on the one hand and other items for $S_1 \parallel S_2$ on the other hand, are synctatically identical. Thus (69) follows.

D.1.2 Statement 2 of Theorem 5: from Mixed Automata to SPA

Proof Consider the following reverse mapping $M \mapsto P_M$, from Mixed Automata to SPA:

1. The alphabet Σ of P_M is identical to that of M;
2. The set of states Q of P_M is equal to the set of states of M, namely the domain of its set X of variables;
3. For $S = (\Omega, \pi, X, p, C)$, decompose relation $\{(\omega, q) | \omega C q\}$ as $\bigcup_{\psi \in \Psi_C} \text{graph}(\psi)$, where Ψ_C denotes the set of all partial functions $\Omega \to Q$, mapping each $\omega \in \exists q.C$ to some q such that $\omega C q$. Then, we consider, for each $\psi \in \Psi_C$, the measure defined by $\psi[\pi](q) =_{\text{def}} \pi(\psi^{-1}(q))$, where $\psi^{-1}(q) = \{\omega | \psi(\omega) = q\}$ ($\psi[\pi]$ is the image of π by ψ), and we renormalize it by considering
$$\frac{\psi[\pi]}{\psi[\pi](Q)}\,,$$
thus obtaining a probability distribution over Q. This defines a subset $\mathbf{P}_S \subseteq \mathcal{P}(Q)$ of probability distributions.
4. The transition relation of P_M is defined as follows:

$$\rightarrow_{P_M} = \{(p, \alpha, \mu) \mid \exists S : (p, \alpha, S) \in \rightarrow_M \text{ and } \mu \in \mathbf{P}_S\} \tag{70}$$

Consider two Mixed Automata M, M' and let $\leq$ be a simulation relation between their state spaces Q and Q': $q \leq q'$ and $q \xrightarrow{\alpha}_M S$ imply the existence of $S' \in \mathbb{S}(Q')$ such that $S \leq^{\mathbb{S}} S'$ and $q' \xrightarrow{\alpha}_{M'} S'$. We need to show that the same relation $\leq \subseteq Q \times Q'$ is also a simulation relation for SPA. Let $q \xrightarrow{\alpha}_{P_M} \mu$ be a transition of SPA P_M. By (70), there exists a Mixed System S such that $q \xrightarrow{\alpha}_M S$ and $\mu \in \mathbf{P}_S$. Since $\leq$ is a simulation relation for Mixed Automata, there exists $S' \in \mathbb{S}(Q')$ such that $S \leq^{\mathbb{S}} S'$ and $q' \xrightarrow{\alpha}_{M'} S'$. Now, $S \leq^{\mathbb{S}} S'$ expands as follows: There exists a weighting function $w : \Omega \times \Omega' \to [0, 1]$ such that the following two conditions hold:

1. For every triple $(\omega, \omega'; q)$ such that $w(\omega, \omega') > 0$ and $\omega C q$, there exists q' such that $\omega' C' q'$ and $q \leq q'$;
2. w projects to π and π', respectively.

Let $\psi \in \Psi_C$ be the selection function giving rise to μ following step 3, meaning that μ is obtained by renormalizing $\psi[\pi]$. Select any $\omega \in \exists q.C$ and let $q = \psi(\omega)$. Select any ω' such that $w(\omega, \omega') > 0$ and assign to it one q' such that $\omega' C' q'$ and $q \leq q'$ (such an q' exists by the above Condition 1). This selection procedure defines a selection function $\psi' : \exists q' C' \to Q'$, mapping the ω' of the above Condition 1 to q', which in turn defines a probability distribution μ', obtained by renormalizing $\psi'[\pi']$. Consider the following weighting function over $Q \times Q'$:

$$v = (\psi, \psi').w\,, \text{ which expands as}$$
$$v(q, q') = w\{(\hat{\omega}, \hat{\omega}') \mid \psi(\hat{\omega}) = q, \psi'(\hat{\omega}') = q'\}$$

In particular $v(q,q') \geq w(\omega,\omega') > 0$ by construction of ψ, ψ', and v. Then, v projects to μ, and to μ':

$$\begin{aligned}\forall q : \sum_{q'} v(q,q') &= \sum_{q'} w\{(\hat{\omega},\hat{\omega}') \mid \psi(\hat{\omega}) = q, \psi'(\hat{\omega}') = q'\} \\ &= \sum_{\omega'} w\{(\hat{\omega},\hat{\omega}') \mid \psi(\hat{\omega}) = q\} = \mu(q)\end{aligned}$$

and

$$\begin{aligned}\forall q' : \sum_{q} v(q,q') &= \sum_{q} w\{(\hat{\omega},\hat{\omega}') \mid \psi(\hat{\omega}) = q, \psi'(\hat{\omega}') = q'\} \\ &= \sum_{\omega} w\{(\hat{\omega},\hat{\omega}') \mid \psi'(\hat{\omega}') = q'\} = \mu'(q')\end{aligned}$$

To summarize, we have constructed a probability distribution μ' such that $\mu \leq^{\mathcal{P}} \mu'$ and $q' \xrightarrow{\alpha}_{P_{M'}} \mu'$, showing that $\leq$ was also a simulation relation for SPA.

To complete our proof, it remains to show the following lemma:

Lemma 6 *There is no mapping $M \mapsto P_M$ that preserves the parallel composition.*

To support the above claim, we consider the following counter-example, where $S(p)$ indicates that S has previous state p:

Example 18 Let $X = \{x_1, x, x_2\}$ be a set of three variables with finite domains Q_{x_1}, Q_x, Q_{x_2}. Consider the two systems $S_i(p_i) = (\Omega_i, \pi_i, X_i, p_i, C_i), i = 1,2$, where: $X_1 = \{x_1, x\}$, $X_2 = \{x, x_2\}$; $p_1 \bowtie p_2$; $\Omega_i = Q_i$ with $Q_1 = Q_{x_1} \times Q_x$ and $Q_2 = Q_x \times Q_{x_2}$; π_i is a probability over Ω_i; and $\omega_i C_i q_i$ iff $\omega_i = q_i$. Define

$$\mathbf{x} : \Omega_1 \uplus \Omega_2 \to Q_x, \text{ such that } \begin{cases} \mathbf{x}(\omega_1) = q \text{ if } \omega_1 = (q_1, q) \\ \mathbf{x}(\omega_2) = q' \text{ if } \omega_2 = (q', q_2) \end{cases} \tag{71}$$

System S_1 amounts to defining the pair (x_1, x) as random variables with joint distribution π_1; similarly, S_2 amounts to defining the pair (x, x_2) as random variables with joint distribution π_2. We assume that the set of all $q \in Q_x$ such that $\pi_1(Q_1 \times \{q\}) > 0$ and $\pi_2(\{q\} \times Q_2) > 0$ is non empty. Forming the composition $S_1 \parallel S_2$ yields the system $S(p) = (\Omega, \pi, X, p, C)$, where $X = X_1 \cup X_2 = \{x_1, x, x_2\}$, $Q = Q_{x_1} \times Q_x \times Q_{x_2}$, $p = p_1 \sqcup p_2$, $\Omega = \Omega_1 \times \Omega_2$, $\pi = \pi_1 \times \pi_2$, and $\omega C(q_1, q, q_2)$ iff $\omega_1 C_1(q_1, q)$ and $\omega_2 C_2(q, q_2)$. According to Definition 1, the sampling of S is the following: draw (ω_1, ω_2) at random with the conditional distribution $\pi_1 \times \pi_2\big((\omega_1, \omega_2) | \mathbf{x}(\omega_1) = \mathbf{x}(\omega_2)\big)$, where the map $\mathbf{x}$ was defined in (71); the resulting (ω_1, ω_2) uniquely defines $(q_1, q, q_2) \in Q$ (no nondeterminism). In words, the parallel composition $S_1 \parallel S_2$ amounts to making the triple of variables (x_1, x, x_2) to be random with the joint distribution $\pi_1 \times \pi_2\big((\omega_1, \omega_2) \mid \mathbf{x}(\omega_1) = \mathbf{x}(\omega_2)\big)$.

Next, consider the Mixed Automaton $M = (\{\alpha\}, X, q_0, \to)$, where $X = \{x_1, x, x_2\}$, set Q of states is defined accordingly $Q = Q_{x_1} \times Q_x \times Q_{x_2}$, and $\to$ maps, through action α, any state $p \in Q$ to the above system $S(p)$. Similarly, we consider the two Mixed Automata $M_i = (\{\alpha\}, X_i, q_{i,0}, \to_i), i = 1,2$, where X_i is as above, $q_{i,0}$ is the projection of q_0 on Q_i, and $\to_i$ maps, through action α, any state $p_i \in Q_i$ to the above system $S_i(p_i)$. We have $M = M_1 \parallel M_2$.

The only candidate way of mapping M_i to a SPA is by considering the two SPA P_i with sets of states Q_i and transition relation $p_i \xrightarrow{\alpha}_i \pi_i$, where π_i was defined above. Now, $P_1 \parallel P_2$ has transition relation $p \xrightarrow{\alpha} \pi_1 \times \pi_2$, which reflects no interaction between the two SPA, so it cannot represent $M_1 \parallel M_2$.

D.2 Proof of Theorem 6 regarding Probabilistic Automata

Proof We consider the mapping $P \mapsto M_P = (\{1\}, X, q_0, \rightarrow_{M_P})$, from PA to Mixed Automata, defined as follows:

1. Alphabet $\{1\}$ is the trivial singleton (the particular element does not matter);
2. $X = \{\xi_\Sigma, \xi_Q\}$, where the variables ξ_Σ and ξ_Q enumerate Σ and Q;
3. Transition $\rightarrow_{M_P}$ maps state p to system $S(p) = ((\Omega, \pi), X, p, C)$, where
   - $\Omega = (\Sigma \times Q)^n$, where n is the cardinal of the image of p by transition $\rightarrow$;
   - π is the product of all the distributions selected by transition $\rightarrow$ starting from p;
   - C is the nondeterministic selection of one component of ω.

We only need to prove the positive statement related to simulation. Consider a simulation relation for PA $q \leq q'$. We need to prove that $\leq$ is also a simulation relation for Mixed Automata. Let μ be such that $(q, \mu) \in \rightarrow$. Since $\leq$ is a simulation relation for PA, there exists μ' such that $(q', \mu') \in \rightarrow'$ and $\mu \leq^{\mathcal{P}} \mu'$. Let S and S' be the mixed systems to which q and q' are mapped by step 3 of the mapping $P \mapsto M_P$. We have to prove that $S \leq^{\mathbb{S}} S'$. For each μ such that $(q, \mu) \in \rightarrow$, let the function $\chi : \mathcal{P}(Q) \rightarrow \mathcal{P}(Q')$ select one μ' such that $(q', \mu') \in \rightarrow'$ and $\mu \leq^{\mathcal{P}} \mu'$ and let v_μ be a weighting function associated to relation $\mu \leq^{\mathcal{P}} \mu'$. The following weighting function

$$w(\omega, \omega') =_{\text{def}} \prod_{\mu : (q,\mu) \in \rightarrow} v_\mu(q_\mu, q'_\mu)$$

where $(q_\mu, q'_\mu) \in Q \times Q'$, solves the problem.