Mixed Nondeterministic-Probabilistic Automata: Blending graphical probabilistic models with nondeterminism

Albert Benveniste, Jean-Baptiste Raclet

Mixed Nondeterministic-Probabilistic Automata: Blending graphical probabilistic models with nondeterminism

Albert Benveniste, Jean-Baptiste Raclet

RESEARCH REPORT
N° 9447
January 2022
Project-Team Hycomes

ISSN 0249-6399 ISRN INRIA/RR--9447--FR+ENG

Mixed Nondeterministic-Probabilistic Automata: Blending graphical probabilistic models with nondeterminism

Albert Benveniste*, Jean-Baptiste Raclet†

Project-Team Hycomes

Research Report n° 9447 — January 2022 — 52 pages

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. In this paper we develop a new model of mixed (nondeterministic/probabilistic) automata that subsumes both nondeterministic automata and graphical probabilistic models. Mixed Automata are equipped with parallel composition, simulation relation, and support message passing algorithms inherited from graphical probabilistic models. Segala's Probabilistic Automata can be mapped to Mixed Automata.

Key-words: factor graphs, Bayesian networks, nondeterminism, probabilistic automata, probabilistic programming

* Inria Rennes, Campus de Beaulieu, 35042 Rennes cedex, France; e-mail: prenom.nom@inria.fr
† IRIT, Université de Toulouse, Toulouse, France; e-mail: Jean-Baptiste.Raclet@irit.fr

RESEARCH CENTRE
RENNES – BRETAGNE ATLANTIQUE

Campus universitaire de Beaulieu
35042 Rennes Cedex

Automates Hybrides Nondéterministes-Probabilistes: mélanger modèles probabilistes graphiques avec le nondéterminisme

Résumé : Les modèles graphiques sont apparus comme utiles depuis les années 1990, dans le domaine des statistiques et de la programmation probabiliste—on regroupe sous ce terme les graphes factoriels et le réseaux Bayésiens. Dans ce rapport on présente le nouveau modèle des *systèmes* et *automates mixtes* (probabilistes/nondéterministes). Ces modèles étendent à la fois les modèles graphiques et les automates probabilistes à la Segala-Lynch.

Mots-clés : graphes factoriels, réseaux Bayésiens, nondéterminisme, automates probabilistes, programmation probabiliste

Contents

1 Introduction

1.1 Context

Bayesian *graphical modeling* and inference [27] expanded since the 1980's, with applications in numerous areas. Graphical models were introduced in probability and statistics to allow for a modular description of models [65]. Graphical models divide into two subfamilies: (directed) Bayesian Networks originally proposed by Judea Pearl [52] and (nondirected) Factor Graphs [43, 46, 65]. Probabilistic graphical modeling gave birth to an important sub-community of probabilistic programming [47, 54, 65].

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 an unnormalized probability $p_f(V_f)$ for the tuple 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 [43].

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. As pointed out by Judea Pearl [53], causality is an extra information relating random variables, not inferrable from their joint probability distribution. *Message passing* algorithms are a key tool for Factor Graphs, allowing to map a subclass of them to Bayesian networks, see [46] and Section 3.6 of [65]. Through the union of underlying graphs and the compositional nature of probabilities specified by graphical probabilistic models, both frameworks of Bayesian Networks and Factor Graphs are naturally equipped with some kind of parallel composition. All these features explain why graphical models are considered as an intermediate format targeted by some probabilistic programming tools, see, e.g., [47, 54], and [65], chapter 3.

One important issue 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 [45]. Also, the mixing of probabilistic behaviors and nondeterminism is central in probabilistic programming [49, 21, 50]. How to blend probabilistic and nondeterministic behaviors in general is, therefore, an important issue.

Probabilistic Programming [47, 54, 20, 51, 42, 65, 9] provides support for specifying statistical models with modularity and libraries for performing inference. Some probabilistic languages generate *likelihood functions* [54, 47, 20] for use by inference algorithms, whereas other generate *sampling* procedures [33, 34]. 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. Objectives of Probabilistic Programming can be categorized as follows:

1. *Modeling paradigm.* Blending probability and nondeterminism, composing, 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 stationarity (or time invariance) of the underlying probabilistic model. 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 the blending of nondeterminism with probabilities, as it generally breaks the stationarity of the underlying statistical model. For example, if two different statistical models are combined with a nondeterministic choice (or an if-then-else statement with nondeterministic guard), then stationarity of the overall model no longer holds. The solution is to recover stationary models by separating the two alternatives and not mixing them. This quickly becomes cumbersome if several such constructions are used in a model. This difficult and central issue is extensively discussed in [39], where it is shown that some major probabilistic programming tools may not correctly implement Monte-Carlo based learning algorithms such as Metropolis-Hastings.

1.2 Contribution

To illustrate our purpose, we begin with a toy example. Throughout this paper, all variables possess finite or denumerable type—this restriction is motivated by technical reasons explained later. Hence, types will not be declared when presenting examples. In "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):

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

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 $\{x_k \mid k{\in}\mathbb{N}\}$. In (1), f_n is a boolean signal indicating the occurrence of a failure and v_n is a noise, i.e., some kind of 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) looks like a dynamical system as usual, with inputs u, f, and v, state x, and output y.

We are interested in a different interpretation, however, by which model (1) specifies what is observed/unobserved: u_n is observed at every instant (as stated in the first line), whereas other signals are unobserved (this is 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:

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

where $v_n \sim \mu$ means that variable v_n has distribution μ at each instant n. As an important 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 $S_1 \| S_2$, defined as the conjunction of the two systems of equations (1,2). $S_1 \| S_2$ combines stochastic behavior with nondeterminism (since failure signal f_n is still unknown and unobverved). 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:

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

In this model, "root failure" signal rf is modeled as a Bernoulli sequence, i.e., $P(rf = \mathtt{T}) = 10^{-6}$; boolean signal bk indicates that a "backup sensor" is provided. Thus, a failure is raised ($f = \mathtt{T}$) if a root failure occurs, and it remains subsequently raised, until a backup sensor is provided. 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 $S_1 \| S_2$ with S_3. By convention of the parallel composition, as a consequence of composing the two statements "$v_n \sim \mu$" and "$rf_n \sim \text{Bernoulli}(10^{-6})$", *the two random sequences v_n and rf_n are 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, 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 occured, 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". It cannot be measured by a probability, since $S_1 \| S_2 \| S_3$ mixes probability with nondeterminism. By "risk" we mean a pessemistic evaluation of this probability, with nondeterminism acting as an adversary.

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

$$S_4 \quad : \quad \texttt{observe}\ y \tag{5}$$

where no dynamics is otherwise specified. Parallel composition $S_1 \| S_2 \| S_3 \| S_4$ expands as the following model:

$$\begin{cases} \texttt{observe}\ u, y \\ x_0 = c_x\ ,\ v_0 = c_v\ ,\ f_0 = \mathtt{F} \\ x_n = \varphi(u_n, x_{n-1}) \\ y_n = \text{if } f_n \text{ then } \psi(x_n, v_n) \text{ else } x_n \\ f_n = (rf_n \text{ or } f_{n-1}) \text{ and not } bk_n \end{cases} \tag{6}$$

$$\begin{cases} rf_n \sim \text{Bernoulli}(10^{-6}) \\ v_n \sim \mu \\ (rf_n \text{ and } v_n \text{ are mutually independent i.i.d. signals}) \end{cases} \tag{7}$$

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 tuple of variables involved in the system. The `observe` constraint on the pair u, y states that its joint trajectory is given (through the sensors). As a consequence, the pair (v, rf) of random signals is now equipped with the *posterior distribution* resulting from constraint (6) being enforced.

If we regard systems $S_1, \ldots, S_4$ as boxes with wires (the involved signals), this modeling approach naturally leads to graphical models alike 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 [36] (born from concurrent constraint programming).

As a semantic domain for the above modeling approach, we propose a framework subsuming graphical probabilistic modeling and supporting both probabilistic and nondeterministic behaviors. We focus our effort on semantics issues, such as: What is actually the probabilistic model specified? Given seemingly different system specifications, are they equivalent or do they differ? Can one define a parallel composition of models? *With reference to the context of probabilistic programming recalled in Section 1.1, our work focuses on objective 1 only, with no consideration of other objectives.*

One of our contributions is the model of *Mixed Automata*. Its design relies on a very simple idea. An automaton is specified through its set of transitions $q \xrightarrow{\alpha} q'$, where q and q' are the current and next state, and α is the action triggering the transition. Upgrading this model to Probabilistic Automata [48] consists in upgrading transitions to $q \xrightarrow{\alpha} \pi'$, where π' is the next probabilistic state (a probability distribution over the set Q of states), from which the next state is derived by probabilistic sampling $\pi' \rightsquigarrow q'$. The final upgrade to Mixed Automata is by upgrading such transitions to $q \xrightarrow{\alpha} S'$, where S' is now a *Mixed System* (or Mixed Probabilistic/Nondeterministic System in its extended name), from which the next state is derived by sampling $S' \rightsquigarrow q'$.

Initially proposed in [13], Mixed Systems 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. This allows to equip Mixed Systems with a parallel composition, on top of which a parallel composition for Mixed Automata can be defined. We show that Mixed Systems naturally inherit a notion of *graphical structure*, which subsumes both Bayesian Networks and Factor Graphs. Mixed Systems offer the counterpart of angelic/demonic nondeterminism [21] and hard/soft conditioning [61, 64], which are important notions in probabilistic programming.

Mixed Automata, defined on top of Mixed Systems, naturally inherit their associated graphical structure and parallel composition. Mixed Automata are equipped with all the fundamental modular notions for Automata, namely the notions of (bi)simulation relation and parallel composition. In this paper, we show in addition that Mixed Automata subsume, in part, Segala's Probabilistic Automata (PA) [60] and their variants. More precisely, we exhibit mappings from different PA models to Mixed Automata, preserving simulation equivalence. The parallel compositions, however, are most of the time different—we claim ours to be more useful than PA parallel composition when the two differ. In addition, in contrast to PA, our model of Mixed Au-

tomata naturally captures the notion of posterior (conditional) distribution and offers a notion of graphical model.

The paper is organized as follows. Mixed Systems are introduced and further studied in Section 2. Mixed Automata are introduced and studied in Section 3, and then compared to Probabilistic Automata in Section 4. Related work is discussed more broadly in Section 5. 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; so we felt it useful to relate them. Finally, Appendix D presents hints for extending our approach to continuous probability distributions.

2 Mixed Probabilistic/Nondeterministic Systems

$\mathcal{X}$ 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. Whenever convenient, we regard X, Y, Z as tuples. 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, i.e., 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 [25], whereas conditional distributions require additional topological assumptions for their existence, and so does the notion of support. To keep our work simpler, we decided not to cover those extensions. Appendix D presents hints for extending our approach to continuous probability distributions.

2.1 Mixed Systems, parallel composition, and Factor Graphs

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].

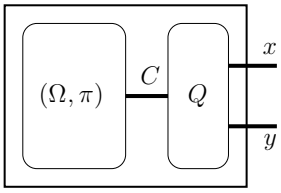

Figure 1: Intuitive picturing of a Mixed System having two variables x and y.

The intuition is illustrated on Figure 1, which will guide us for 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 tuple 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 private/visible is shown on Figure 1 by the outgoing pins x, y, which contrast with the absence of outgoing pin for the probabilistic box.

We are interested in understanding how Mixed Systems are "executed" (we call this the *sampling*), and how state properties—which are not by themselves random—can still get some kind of probabilistic evaluation.

Definition 1 (Mixed System, definition and semantics)

1. *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 $(\omega, q) \in C$*.*

2. S *is called* consistent *if* $\pi(\Omega^{\mathsf{c}}) > 0$*, where* $\Omega^{\mathsf{c}} =_{\text{def}} \{\omega \in \Omega \mid \exists q : \omega C q\}$*. If* S *is consistent, its* sampling *is well defined and consists in:*

   (a) *sampling* $\omega \in \Omega$ *according to conditional probability* π^{c}*, where:*
   $$\forall A \subseteq \Omega \quad : \quad \pi^{\mathsf{c}}(A) =_{\text{def}} \pi(A \mid \Omega^{\mathsf{c}}) = \tfrac{\pi(A \cap \Omega^{\mathsf{c}})}{\pi(\Omega^{\mathsf{c}})}\,, \tag{8}$$

   (b) *and, then, nondeterministically selecting* $q \in Q$ *such that* $\omega C q$*.*

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

3. *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* $A \subseteq Q$*:*
   $$\overline{\pi}(A) =_{\text{def}} \pi^{\mathsf{c}}(\Omega_{\exists A}) \quad \textit{where} \quad \Omega_{\exists A} =_{\text{def}} \{\omega \in \Omega \mid \exists q \in A : \omega C q\}\,, \tag{9}$$
   $$\underline{\pi}(A) =_{\text{def}} \pi^{\mathsf{c}}(\Omega_{\forall A}) \quad \textit{where} \quad \Omega_{\forall A} =_{\text{def}} \{\omega \in \Omega \mid \forall q \in A : \omega C q\}\,, \tag{10}$$

   The following generalized likelihood $\ell : 2^Q \to [0,1]$ *is also of interest:*
   $$\ell(A) =_{\text{def}} \max_{\omega \in \Omega_{\exists A}} \pi^{\mathsf{c}}(\omega)\,. \tag{11}$$

In the sequel, we shall denote by $\mathbb{S}(X)$ *the class of all (possibly inconsistent) Mixed Systems having* X *as their set of variables.* □

$\overline{\pi}$ defined by formula (9) 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

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

Note that (11) resembles (9) if we rewrite the latter as $\overline{\pi}(A) = \sum_{\omega \in \Omega_{\exists A}} \pi^{\mathsf{c}}(\omega)$. The same comments hold, mutatis mutandis, regarding the *inner probability* $\underline{\pi}$, which is super-additive. Note that,

$$\text{if } A = \{q\} \text{ is a singleton, then } \overline{\pi}(A) = \underline{\pi}(A). \tag{12}$$

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

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, simply 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 $q \in \widehat{C}$. □

Example 2 [Specializing to pure probabilistic systems] A pure probabilistic system is specified as a pair (Ω, π). To reformulate 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. □

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 [26, 27, 59].[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., [28, 49, 8, 50, 42, 51, 21, 66]. Nondeterministic choice between alternatives is considered in [50] 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 [21], where *demonic* and *angelic* nondeterminisms are seen as adversarial and beneficial, respectively. These notions mirror the outer and inner probabilities used in Dempster theory. Unfortunately, outer and inner probabilities do not bring limit theorems of probability theory (law of large numbers, etc.), which are the core of machine learning.

Through formulas (9,10) 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. 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 [21].

Finally, the generalized likelihood of formula (11) is the basis for inference, estimation, or machine learning, when multiple hypotheses or nuisance parameters are considered [45]—we are not aware of any use of a mirror notion where "min" would be substituted for "max". □

Example 3 [outer probabilities] Consider model $S_1 \| S_2 \| S_3$ of Section 1.2. 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. Denoting by Q_v the domain of v and by $\mathbb{B}$ the Boolean domain, the underlying probability space of S is (Ω, π), where $\Omega = Q_v \times \mathbb{B}$ and $\pi = \mu \times \beta$, where β is Bernoulli(10^{-6}). Domain Q for the variables of S is $Q = Q_x \times Q_y \times Q_u \times Q_v \times \mathbb{B} \times \mathbb{B}$, and relation C is defined by the nonprobabilistic equations of S, i.e., (1,3) in which we discard the statement $rf_n \sim \beta$. Finally, let $C(u_n)$ denote the relation C

[2] Outer and inner probabilities were called upper and lower in [26].

in which the value of u_n is given (u is observed). Then

$$\begin{array}{l}\overline{\pi}(x_n{>}y_{\max} \text{ and } y_n{\leq}y_{\max}) = \pi^{\mathsf{c}}(W)\ , \quad \text{where} \\ W = \left\{ (v, rf) \;\middle|\; \exists x_n, y_n, f_n, bk_n : \begin{array}{l} x_n{>}y_{\max} \text{ and } y_n{\leq}y_{\max}\ , \text{ and} \\ (x_n, y_n, f_n, bk_n, v, rf) \in C(u_n) \end{array} \right\}\end{array} \tag{13}$$

Inspecting (1,3) shows that the condition defining set W rewrites as

$$x_n{>}y_{\max} \text{ and } \psi(x_n, v){\leq}y_{\max} \text{ and } f_n{=}\mathtt{T} \text{ and } (x_n, y_n, f_n, bk_n, v, rf) \in C(u_n)\,.$$

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

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

Condition $f_n{=}\mathtt{T}$ is equivalent to the conjunction of the following two conditions: 1) $bk_n{=}\mathtt{F}$ (backup sensor is not available), 2) $f_{n-1}{=}\mathtt{T}$ or $rf_n{=}\mathtt{T}$. Recall that the value of f_{n-1} is given. We thus distinguish the following two cases:

1. $f_{n-1}{=}\mathtt{T}$: then, $f_n{=}\mathtt{T}$ whatever the value of bk_n is, and, using (13):

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

2. $f_{n-1}{=}\mathtt{F}$: then, $f_n{=}\mathtt{T}$ if and only if $rf_n{=}\mathtt{T}$ and $bk_n{=}\mathtt{F}$. Thus, chosing $bk_n{=}\mathtt{F}$ ensures that: $rf_n{=}\mathtt{T}$ and $\psi(\varphi(u_n, x_{n-1}), v) \leq y_{\max}$ together yield $y_n{\leq}y_{\max}$. Alternatively, $bk_n{=}\mathtt{T}$ prevents the condition $y_n{\leq}y_{\max}$ from occurring. By definition of the outer probability (9), we finally get, using (13):

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

which corresponds to the probabilistic evaluation of the predicate "$x_n{>}y_{\max}$ and $y_n{\leq}y_{\max}$" if the nondeterministic alternative $bk_n{=}\mathtt{F}/\mathtt{T}$ is interpreted as demonic [21]. □

Discussion 2 (Conditioning and its variations) Conditioning is generally not considered in the field of probabilistic automata. It is, however, central in probabilistic programming, see, e.g., [51, 22, 64] 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 litterature on probabilistic programming distinguishes between *hard* (also called *deterministic*) and *soft* (also called *stochastic*) conditioning [61, 64]. In the basics of probability theory, however, the only notion is that of *conditional expectation* [25], from which other notions are derived, e.g., conditional probability, transition probability or stochastic kernel, and disintegration (or regular version of conditional expectation). Deriving such notions is straightforward in our case, since we restrict ourselves to discrete probability spaces. We will discuss this further when extending Bayesian networks to Mixed Systems, in Section 2.2.

Discussion 3 (consistency) Inconsistency formalizes self-contradiction, for Mixed Systems. The condition "$\pi(\Omega^{\mathsf{c}}) > 0$" in statement 2 of 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 right definition relies on topological properties. As a consequence, our elementary definition of consistency would no longer apply. This is next illustrated on our running example.

Example 4 [consistency] Consider model (6,7). Statement 2 of Definition 1 defines consistency as the existence of a state q in relation through C with an 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 bicontinuous for every fixed x. 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. With reference to Example 3, the only difference is that a parallel composition with the statement `observe` y was added. So, it still makes sense to consider the two cases 1 and 2 of Example 3. In case 1, we get $W = \{(rf, v) \mid \psi(\varphi(u_n, x_{n-1}), v){=}y_n\}$, whence $(\beta{\times}\mu)(W){=}0$. Deducing inconsistency would be nonsense, however, since the support of μ is $\mathbb{R}$. 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.

2.1.1 Equivalence

In this section, we study equivalence. To this end, we introduce the following operation of compression, on top of which equivalence is defined:

Definition 2 (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:*

$$(\omega, \omega') \in \sim \quad \text{if and only if:} \quad \forall q \in Q : \omega C q \Leftrightarrow \omega' C q\,. \tag{14}$$

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 (see item 1 of Definition 1 for notation C_π):

Definition 3 (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 $(\omega', q') =_{\text{def}} \varphi(\omega, q)$:*

$$\omega\, C_\pi\, q \Leftrightarrow \omega'\, C'_{\pi'}\, q' \quad ; \quad \pi'(\omega') = \pi(\omega) \quad ; \quad q' = q\,. \tag{15}$$

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 4 *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,

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

A similar proof holds for inner probabilities. □

Discussion 4 (equivalence) Floyd/Hoare/Dijkstra logic of pre- and postconditions for imperative languages was extended to encompass probability and nondeterminism with pGCL (probabilistic Guarded Command Language) [44, 22, 49, 42, 51, 50]. 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 equivalence of probabilistic programs, see, e.g., Section 3.1 of [65].

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

2.1.2 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 tuple X, $Y \subseteq X$ a subset of variables, and

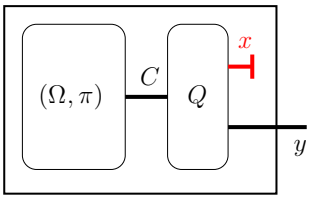

Figure 2: The marginal on y for the Mixed System of Figure 1 is by hiding x (in red).

$Z = X - Y$, 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 5 (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 5 may require a compression.

Example 5 [Link with the classical notion of marginal for probabilities] Let us apply Definition 5 to the purely probabilistic system of Example 2, namely $S_{\mathsf{proba}} = (Q, \pi, \{X, Y\}, diag)$, having two variables X, Y, corresponding state space Q, and $\Omega = Q$ with $C = diag$, the diagonal. This is the model of a pair (X, Y) of visible variables with joint probability distribution $\pi(x, y)$, where x and y denote values for X and Y, respectively. The projection of $diag$ on Y is

$$\mathbf{Pr}_Y(diag) = \{(x, y, y') \mid y = y'\}\,.$$

Thus, $(x, y) \sim (x', y')$ if and only if $y = y'$. Thus, when using the formula of Definition 5 to define $\mathbf{Margin}_Y(S_{\mathsf{proba}})$, the private probability space (Q, π) must be compressed as $\widetilde{\pi}(y) = \sum_x \pi(x, y)$, showing that our notion of marginal boils down to the classical notion for probabilities in this case. □

2.1.3 Parallel composition

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

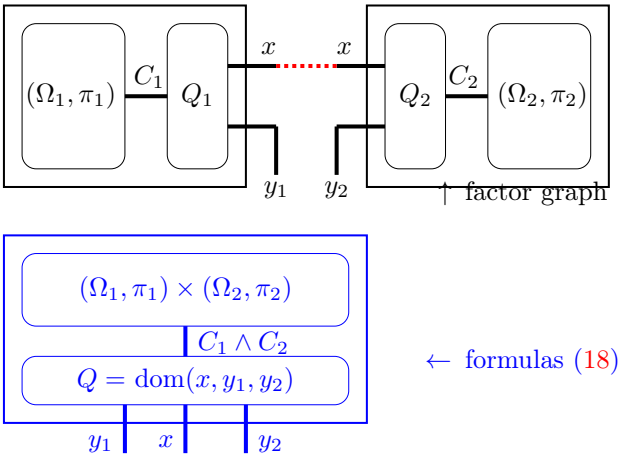

Figure 3: Illustrating the parallel composition, for y_1, y_2 local variables and x shared. The factor graph, capturing the connection via identical wires, is depicted on the top in black; the definition using formulas (18) is shown in blue.

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

$$\bowtie_{i\in I} q_i\,, \tag{16}$$

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*

$$\sqcup_{i\in I} q_i \in Q \tag{17}$$

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

Definition 6 (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:*

$$\begin{array}{rcl} X & = & X_1 \cup X_2 \,,\, \Omega = \Omega_1 \times \Omega_2 \,,\, \pi = \pi_1 \times \pi_2 \quad \textit{(cartesian product), and} \\ C & = & \left\{ \left((\omega_1, \omega_2), q_1 \sqcup q_2 \right) \mid q_1 \bowtie q_2 \;\; \textit{and} \;\; \omega_1 C_1 q_1 \;\; \textit{and} \;\; \omega_2 C_2 q_2 \right\}, \end{array} \tag{18}$$

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 set of vertices collects systems and variables:*

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

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

The composition of two consistent systems may be inconsistent. Let

$$\mathsf{nil} \quad = \quad (\{1\}, \delta_1, \emptyset, \{(1, \epsilon)\}) \tag{19}$$

be the *nil* system, with trivial probability space $(\Omega, \pi) = (\{1\}, \delta_1)$ and no visible variable; its state space is the singleton $Q_{\mathsf{nil}} = \{\epsilon\}$ where ϵ is some distinguished element, 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:

$$\mathcal{G}_{S_1 \parallel S_2} \quad = \quad \mathcal{G}_{S_1} \cup \mathcal{G}_{S_2} \,. \tag{20}$$

The associativity and commutativity of this parallel composition is 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 are useful in decomposing large but sparse systems, into a parallel composition of smaller, locally interacting, subsystems.

Lemma 7 $S_1 \equiv S_1'$ *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.

2.2 Bayesian Calculus and Bayesian Networks

So far Factor Graphs and related algorithms are able to capture joint distributions relating different statistical data, but they cannot capture causality, as argued by Judea Pearl [53]. Actually, 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 [20]), 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 [52]. Bayesian networks specify causality information by means of directed graphs, which bring the extra information advocated by J. Pearl to talk about caussality. Bayesian networks also naturally support incremental execution. In this section, we show how these concepts supporting causality and incremental sampling, can be extended to 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. Sampling $P(x|y)$ consists in 1) nondeterministically selecting a value for y, and then 2) with this value of y, sampling X according to $P(x|y)$. $P(x|y)$ is called a *transition probability,* or a *probability kernel* or *stochastic kernel,* depending on the contexts and communities: $y \mapsto P(x|y)$ maps any value for Y to a probability distribution for X. We now extend these notions to Mixed Systems.

2.2.1 Mixed Kernel

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 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.

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 pair of maps*

$$q_{\text{in}} \mapsto \big(\overline{\pi}(q_{\text{in}}), \underline{\pi}(q_{\text{in}})\big) \tag{21}$$

where q_{in} is a value for the input variables X_K^{in}, and $\overline{\pi}(q_{\text{in}})$ and $\underline{\pi}(q_{\text{in}})$ are the outer and inner probabilities associated to Mixed System $K(q_{\text{in}})$. □

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

$$C_q \quad =_{\text{def}} \quad \{\omega{\in}\Omega \mid \omega C q\}, \text{ and } C_\omega =_{\text{def}} \{q{\in}Q \mid \omega C q\}\,. \tag{22}$$

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

2.2.2 Bayesian Network

Definition 9 (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* Bayesian Network *if satisfies the following conditions:*

$$\forall K \in \mathbb{K} \quad \Longrightarrow \quad X_K^{\text{in}} \subseteq {}^\bullet K \text{ and } X_K^{\text{out}} = K^\bullet\,. \tag{23}$$

$$\forall K_1, K_2 \in \mathbb{K}, K_1 \neq K_2 \quad \Longrightarrow \quad {K_1}^\bullet \cap {K_2}^\bullet = \emptyset \tag{24}$$

For convenience, we will denote by

$$K_1; K_2 \tag{25}$$

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* $X = X^{\text{in}}_{K_1}\cup X^{\text{out}}_{K_1}\cup X^{\text{in}}_{K_2}\cup X^{\text{out}}_{K_2}$*.* □

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

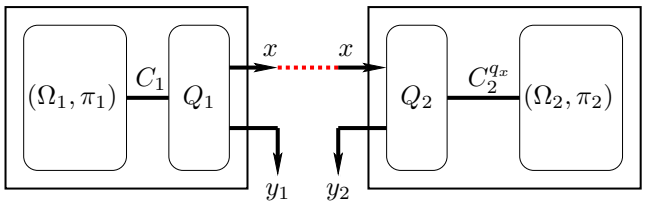

Figure 4: Bayesian Network $S_1; K_2$. Mixed Kernel K_2 has input x.

To Bayesian Network $\mathcal{N}=(X\cup\mathbb{K},\hookrightarrow)$, we associate the partial order $(X\cup\mathbb{K},\preceq)$, where $\preceq$ is the transitive closure of $\hookrightarrow$. In the following definition, for q a valuation of the set X of variables and K a kernel belonging to $\mathbb{K}$, $q_{\downarrow {}^\bullet K}$ and $q_{\downarrow K^\bullet}$ denote 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 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 to every* $q\in Q_X$ *such that* $\mathcal{N} \rightsquigarrow q$*, its* probabilistic score

$$\overline{\pi}(q) \quad = \quad \prod_{K\in\mathbb{K}} \overline{\pi}\big(K, q_{\downarrow {}^\bullet K}\big)\big(q_{\downarrow K^\bullet}\big)\,. \tag{26}$$

In (26), $\overline{\pi}(K, q_{\downarrow {}^\bullet K})(q_{\downarrow K^\bullet})$ *is the score assigned to state* $q_{\downarrow K^\bullet}$ *by the outer probability associated to mixed system* $K(q_{\downarrow {}^\bullet K})$*.* □

Since inclusion $X_- \cup \mathbb{K}^{*\bullet} \supset X_-$ in step 3c is strict, the inductive procedure terminates in finitely many steps. Note that, by (12), there is no need to consider $\underline{\pi}(q)$. The inductive procedure of Definition 10 is formalized in Algorithm 1.

Algorithm 1 Incremental sampling of 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$(K(q_{\downarrow {}^\bullet K}))$
    end for
  end while
  $\mathbb{K}_- \leftarrow \mathbb{K}_- \cup \mathbb{K}^*$
  $X_- \leftarrow X_- \cup \mathbb{K}^{*\bullet}$

Definition 11 (Bayesian network equivalence) *Let $\mathcal{N}_1$ and $\mathcal{N}_2$ be two 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 4, $S \equiv S'$ implies $S \equiv_P S'$, when regarding mixed systems S and S' as Bayesian networks.

Example 6 [Finite Markov chain as a 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 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 Bayesian Networks proposed in Definition 9. □

We next extend, to mixed systems, the notion of conditional distribution. To this end, we will use the following notation: for Y a set of variables and $q_Y \in Q_Y$,

$$(Y{=}q_Y) \tag{27}$$

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 $\mathbf{Cond}_Y(S)\,(q_Y) =_{\text{def}} (Y{=}q_Y) \,\|\, S$.* □

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 tuple 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 $(Y{=}q_Y) \,\|\, S$ yields the conditional distribution $\pi(q_Z|q_Y)$. Therefore, Definitions 5 and 12 extend the notions of marginal and conditional existing on purely probabilistic systems.

Discussion 5 (more on conditioning) When probability and nondeterminism are blended, the notion of Mixed Kernel serves the same purpose as *soft* or *stochastic* conditioning [64], since it implements the stochastic conditioning $p(x \mid y \sim D)$ discussed in the introduction of [64]. □

Generally, sampling the parallel composition $S_1 \parallel S_2$ yields a result which 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 3 regarding isomorphic samplings):

Theorem 13 (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]

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

Proof: See Appendix A.3 for the proof. □

The right hand side of Bayes' formula is illustrated on Figure 5.

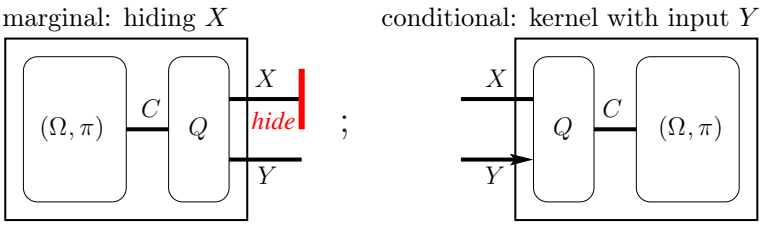

Figure 5: Illustrating the right hand side of Bayes' formula: the output Y of the system on the left is connected to the input Y of the kernel on the right.

By Definition 6, 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 to a parallel composition of several Mixed Systems, to Bayesian Networks while preserving the sampling. This provides such Factor Graphs with an incremental sampling:

Theorem 14 (message passing algorithm) *If Factor Graph $\mathcal{G}_{\mathcal{S}}$ of system $\mathcal{S}$ is a tree, we can transform it to a Bayesian Network $\mathcal{N}_{\mathcal{S}}$ while preserving its probabilistic semantics.*

See Appendix A.4 for a proof.

Message passing algorithms for computing generalized likelihoods. The purpose of probabilistic languages [47, 20, 35] 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 [41, 56]. Theorem 14 shows that message passing algorithms also allow for an incremental evaluation of generalized likelihoods.

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

2.3 The ReactiveBayes minilanguage

In this section, we use the model of Mixed Systems to specify the semantics of the informal language we used in the introduction when discussing our running example. To make this precise, we formalize this informal language through the "ReactiveBayes" syntax presented hereafter.

To prevent from 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.

2.3.1 Syntax

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

$$\begin{array}{rcl} e & ::= & c \mid x \mid (e,e) \mid op(e) \mid f(e) \mid \texttt{pre}\ x \mid \texttt{init}\ x = c \\ S & ::= & x \sim P(e) \mid e = e \mid \texttt{observe}\ x \mid\ S \parallel S \end{array} \tag{28}$$

- 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 Mixed System 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 thereof. For each term P we assume a semantics denoted by π_P, which is a probability.

No provision is given by syntax (28) for writing equations relating systems. In particular, fixpoint equations $S = S' \parallel S$ 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 3.2, where the semantics of full ReactiveBayes will be given.

Example **1** Mixed System S_1, specified by model (1) writes

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

Mixed System S_2, specified by model (2) writes

```
  init noise = n0
|| noise = chi(pre noise,w)
|| w ~ mu
```

And so on. The global model is $S_1 \parallel S_2 \parallel S_3 \parallel S_4$. □

2.3.2 Semantics

We now give the semantics of the static fragment of ReactiveBayes, namely ignoring in (28) the statements `pre` and `init`. $[\![S]\!]$ denotes the semantics of ReactiveBayes program S:

$$\begin{array}{lrcl} (i) & [\![\texttt{observe}\ x]\!] & = & (\cdot,\cdot,\{x\}, x = c) \\ (ii) & [\![x \sim P]\!] & = & (\Omega_x, \pi_P, \{x\}, x = \omega_x) \\ (iii) & [\![x \sim P(e)]\!] & = & c \mapsto [\![x \sim P(c)]\!], \text{ where } c = e \\ (iv) & [\![e = e']\!] & = & (\cdot,\cdot,\text{vars}(e) \cup \text{vars}(e'), e = e') \\ (v) & [\![S_1 \parallel S_2]\!] & = & [\![S_1]\!] \parallel [\![S_2]\!] \end{array} \tag{29}$$

In (i), the semantics has no probabilistic part, and a single visible variable x whose value c 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. In (iii), the probability depends on an expression e, whose generic value is denoted by c; the semantics is the kernel mapping c to $[\![x \sim P(c)]\!]$. Line (iv) defines the semantics of equations; "vars(e)" denotes the set of variables involved in expression e; the semantics has no probabilistic part. Finally, (v) makes this semantics structural. Thanks to formula (20) of Definition 6, it also defines the Factor Graph representing S.

The following fragment of (29) is mapped to a Bayesian Network. In the following formulas, $\mathcal{N}[\![S]\!]$ denotes the Bayesian Network defined by S, when it exists:

$$\begin{array}{lrcl} (i) & \mathcal{N}[\![\texttt{observe}]\!] & = & \{\mathsf{is\_source}(x)\} \\ (ii) & \mathcal{N}[\![x \sim P]\!] & = & \{x\} \\ (iii) & \mathcal{N}[\![x \sim P(e)]\!] & = & \mathrm{vars}(e) \to [\![x \sim P(e)]\!] \to x \\ (iv) & \mathcal{N}[\![x = e]\!] & = & \mathrm{vars}(e) \to [\![x = e]\!] \to x \\ (v) & \mathcal{N}[\![S_1 \parallel S_2]\!] & = & \mathcal{N}[\![S_1]\!] \cup \mathcal{N}[\![S_2]\!] \end{array} \tag{30}$$

In (i), $\mathsf{is\_source}(x)$ denotes x flagged with the condition that it must remain a source node in any of its environments. Application of Rule (v) is subject to the following success conditions:

Condition 1 (success conditions)

1. *The union $\mathcal{N}[\![S_1]\!] \cup \mathcal{N}[\![S_2]\!]$ possesses no circuit and satisfies the conditions of Definition 9, and*
2. *The result keeps satisfying all inherited conditions (i).*

These conditions ensure that parallel compositions are incremental. The message Passing algorithm presented in Theorem 14 allows source-to-source rewriting for mapping tree shaped non-directed Factor Graphs to directed Bayesian Networks.

Example **2** The following picture displays, on the top, the Factor Graph associated to $S_1 \parallel S_2 \parallel S_3 \parallel S_4$, and, on the bottom, the Bayesian Network resulting from applying the message passing algorithm—for better readability we show only shared variables:

$$\begin{array}{c} S_4 \\ | \\ y_n \\ | \\ S_2 - f_n - S_1 - v_n - S_3 \end{array}$$

$$\begin{array}{c} \mathbf{Cond}_y(S_4) \\ \uparrow \\ y_n \\ \uparrow \\ \mathbf{Cond}_f(S_3) \leftarrow f_n \leftarrow \mathbf{S}_1 \rightarrow v_n \rightarrow \mathbf{Cond}_v(S_2) \end{array}$$

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

Discussion 6 (if-then-else) In Example 1, system S_1 involves an "if-then-else" statement. Syntax (28), however, does not involve such statements. This means that "if-then-else" statements are seen by syntax (28) 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 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 definitely needed in probabilistic reasoning [21], see Appendix A.1. The same holds when performing inference or learning [39]; see also the discussion of objective 3 of probabilistic programming on page 5. □

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.

3 Mixed Automata

The idea is simple: 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 [58, 48].
3. Upgrading them further to $q \xrightarrow{\alpha} S' \rightsquigarrow q'$, where S' is a Mixed System and $\rightsquigarrow$ denotes sampling, yields Mixed Automata.

3.1 Definition and properties

The formal definition is introduced next. It uses the notation $\mathbb{S}(X)$, introduced at the end of Definition 1. We assume an underlying alphabet $\mathbf{\Sigma}$ of *actions*.

Definition 15 (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 that* M *shall be* deterministic*:*

$$\text{for any pair } (q, \alpha) \in Q \times \Sigma,\ q \xrightarrow{\alpha} S \text{ and } q \xrightarrow{\alpha} S' \text{ implies } S = S'. \tag{31}$$

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

$$\mathbf{r} \quad = \quad q_0 \xrightarrow{\alpha_1} S_1 \rightsquigarrow q_1 \xrightarrow{\alpha_2} S_2 \rightsquigarrow q_2 \ldots q_{k-1} \xrightarrow{\alpha_k} S_k \rightsquigarrow q_k\,, \tag{32}$$

where Mixed Systems $S_1, \ldots, S_k$ *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 (31).

Example 7 [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 $M = (\Sigma, X, q_0, \to)$, where: Σ is the singleton $\{\alpha\}$; variable X has domain Q; $\to$ 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 (17) and Definition 6, 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 minilanguage, we will, however, need the more flexible synchronization mechanism of "compatible actions"—this is known to be only a technical extension. We thus assume that the underlying alphabet $\mathbf{\Sigma}$ of actions is equipped with a commutative and associative *join* partial operation $\sqcup_{\mathbf{\Sigma}} : \mathbf{\Sigma} \times \mathbf{\Sigma} \to \mathbf{\Sigma}$, where $\alpha_1 \sqcup_{\mathbf{\Sigma}} \alpha_2$ is defined whenever the two actions are *compatible*, written $\alpha_1 \bowtie_{\mathbf{\Sigma}} \alpha_2$. 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 16 (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 6:*

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

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

1. In the context of automata, relation $\leq$ on pairs of states is a *simulation* if it satisfies [57]:

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

2. This definition is upgraded to Probabilistic Automata as follows [57]:

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

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

$$\begin{array}{l} \pi_1' \leq^P \pi_2' \text{ ensures, for each } q_1' \text{ such that } \pi_1' \rightsquigarrow q_1', \\ \text{the existence of } q_2' \text{ satisfying } \pi_2' \rightsquigarrow q_2' \text{ and } q_1' \leq q_2'. \end{array} \tag{33}$$

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

$$\left.\begin{array}{l} q_1 \xrightarrow{\alpha} S_1' \\ q_1 \leq q_2 \end{array}\right\} \implies \exists S_2' : \left\{\begin{array}{l} q_2 \xrightarrow{\alpha} S_2' \\ S_1' \leq^S S_2' \end{array}\right. \tag{34}$$

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

$$\begin{array}{l} S_1' \leq^S S_2' \text{ shall ensure, for each } q_1' \text{ such that } S_1' \rightsquigarrow q_1', \\ \text{the existence of } q_2' \text{ satisfying } S_2' \rightsquigarrow q_2' \text{ and } q_1' \leq q_2'. \end{array} \tag{35}$$

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

Definition 17 (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 (35). 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 $(S_1, S_2) \in \rho^{\mathbb{S}}$.

Discussion 7 (lifting and coupling) Our lifting is a direct extension of the technique used in [57] for Probabilistic Automata. In the context of probabilistic reasoning, the same technique was also extensively studied under the name of *probabilistic coupling* [8, 38]. Weighting function $w(\omega_1, \omega_2)$ of Definition 17 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 *Non-deterministic labelled Markov process* in [23, 29], in a categorical framework; the second reference encompasses continuous distributions (beyond discrete).

Lemma 18 *$S_1 \, \rho^{\mathbb{S}} \, S_2$ and $S_1' \equiv S_1$ together imply $S_1' \, \rho^{\mathbb{S}} \, S_2$.*

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

Definition 19 (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.* $\square$

Discussion 8 (simulation equivalence vs. bisimilarity) Despite the condition (31) 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. Actually, we will prove in our forthcoming Theorem 22 that Segala's Probabilistic Automata [57, 58, 48], which possess nondeterministic transition relations, can be embedded into Mixed Automata while preserving simulations. $\square$

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 20 *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 B.2 for a proof. □

Discussion 9 (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 $(N+1)$-tuple 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 Bayesian Network.

3.2 Mixed Automata for the semantics of ReactiveBayes

In this section we first give the semantics of the full ReactiveBayes minilanguage (28) in terms of Mixed Automata. Recall that the semantics of the static part of the language was given in (29,(i)–(v)).

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

$$ {}^{\bullet(n+1)}x =_{\text{def}} {}^\bullet({}^{\bullet n}x) \quad \text{and} \quad Q_{{}^\bullet x} = Q_x \,. \tag{36} $$

Then, we define

$$ {}^\bullet e(x) \quad =_{\text{def}} \quad e({}^\bullet x) \tag{37} $$

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 (27): this system has trivial probabilistic part, variable x, and enforces the value q_x for it. □

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

$$ \begin{array}{lrcl} (vi) & [\![\texttt{pre}\ x]\!] & = & \left(\{\text{T}\}\,, \{x, {}^\bullet x\}\,, -\,, \{q \xrightarrow{\text{T}} ({}^\bullet x{=}q_x) \mid \forall q \in Q\}\right) \\ (vii) & [\![\texttt{init}\ x = c]\!] & = & \left(\{\text{T}\}\,, \{x\}\,, c\,, c \xrightarrow{\text{T}} \mathsf{nil} \text{ and } \epsilon \xrightarrow{\text{T}} \mathsf{nil}\right) \end{array} \tag{38} $$

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

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

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 initialization statement `init`. Its semantics is stated in (vii), where nil is the trivial Mixed System defined in (19). 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.

So far we have completed the semantics of ReactiveBayes as defined in (28), 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 (28):

$$\begin{array}{lll} \alpha & ::= & {}^{\bullet}e, \text{ where } e \text{ has Boolean type} \\ A & ::= & \texttt{on}\ \alpha\ \texttt{then}\ S\ \texttt{else}\ S \mid\ A \parallel A \end{array} \tag{39}$$

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 ($\top$ denotes the Boolean value "true", and we refer the reader to Definition 1 regarding nil and the distinguished state ϵ):

$$\begin{array}{lll} (viii) & [\![\texttt{on}\ \alpha\ \texttt{then}\ S\ \texttt{else}\ S']\!] & = \dfrac{S \text{ and } S' \text{ have previous state } p}{\left(\begin{array}{c} \{\alpha, \neg\alpha\}\ ,\ X \cup X'\ ,\ \cdot \\ \{p \xrightarrow{\alpha} S, p \xrightarrow{\neg\alpha} S'\} \end{array}\right)} \\ (ix) & [\![A_1 \parallel A_2]\!] & = [\![A_1]\!] \parallel [\![A_2]\!] \end{array} \tag{40}$$

The right hand side of $(viii)$ is an inference rule meaning "*numerator entails denominator*". By (37) and the syntax for actions in (39), 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 16, we now formalize the compatibility relation $\bowtie_{\mathbf{\Sigma}}$ and the join operator $\sqcup_{\mathbf{\Sigma}}$:

$$\alpha_1 \bowtie_{\mathbf{\Sigma}} \alpha_2 \text{ always holds, and } \alpha_1 \sqcup_{\mathbf{\Sigma}} \alpha_2 =_{\text{def}} \alpha_1 \wedge \alpha_2. \tag{41}$$

4 Comparison with Segala's Probabilistic Automata

Probabilistic Automata (PA) [57, 58, 48] were originally proposed by Segala and Lynch. To simplify our comparison, we discuss here the version of PA with no consideration of internal actions. According to the classification made by Sokolova and de Vink [60], 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 [60].

Simple Probabilistic Automata existed way before the work of Segala and Lynch [57, 58, 48], in the community of applied mathematics and probability theory, where they are known under the name of *Markov Decision Processes (MDP)* [10, 55]. In this context, the main considered problem is the synthesis of an *optimal policy* to minimize some expected cost function on 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 [32], and further developed to support robustness by defining metrics between finite MDP's [30].

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:

$$\textit{Simple Probabilistic Automaton (SPA)} \quad : \quad \rightarrow \subseteq Q \times \Sigma \times \mathcal{P}(Q) \tag{42}$$

$$\textit{Probabilistic Automaton (PA)} \quad : \quad \rightarrow \subseteq Q \times \mathcal{P}(\Sigma \times Q) \tag{43}$$

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 17 adapted by ignoring relations C_1 and C_2.

Details for SPA, model (42)

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 [48] 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 $i = 1,2$.

Details for PA, model (43)

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 [60] 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 among the two transitions must be elected while the other one 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). Other variants for solving the same conflict all need such additional probability distributions—typically referred to as *schedulers*.

Discussion 10 (who comes first: nondeterminism or probability?) The following question arises [66]: should nondeterminism be resolved *prior* 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 schyzophrenic approach: actions are selected first, leading to a Mixed System in which nondeterminism is resolved at last (See point 2 in Definition 1)—one can thus say that nondeterminism is resolved "first-and-last". As we shall see in our forthcoming comparison, the main difference between our model and models from the PA family is not in this "prior vs. after" issue, but rather in our handling of conditioning and parallel composition. □

Comparison results

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

Theorem 21 (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 C.1.1 and C.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 22 (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 C.2 for a proof.

Due to Statement 2 of Theorem 21 and the existence of an embedding SPA→PA [60] preserving simulation, a reverse mapping exists, from Mixed Automata to PA.

In [60], 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 → Mixed Automata and PA → 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.

Discussion 11 (More on comparing SPA/PA and Mixed Automata) So far Theorems 21 and 22 compare SPA/PA and Mixed Automata regarding the core notions of PA, namely simulation and parallel composition. Conditioning is not at all considered in PA theories—this indeed is the reason for them to have problems when handling synchronization in the parallel composition.

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. □

5 Other related work

So far we 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 can be produced by the considered system. 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 1970's, 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 [62]. In our context of nondeterministic/probabilistic dynamical systems, the task was not really investigated by mathematicians, and one should rather look at the literature closer to computer science. The seminal paper by Kozen [44] defines two kinds of semantics of simple imperative probabilistic programs. The first semantics has 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 [63, 35]. This approach was extended in [40, 42, 22] in order to provide semantics to the *observe* statement present in most modern probabilistic programming languages. In [16], 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* [20, 65], all of them offer *while loops*. These features raise the issue of possible non termination. Non terminating while loops are the essence of [9]. We did not consider *recursion* in its full generality, but only under the limited form of non-terminating time-recursion, with Mixed Automata. Actually, 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 [17, 39, 20, 33, 34]. 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 [39]. We did not consider learning in this work. Clearly, 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 [31]—this is a different subject than statistical testing in the sense of [45].

In Section 4, we have shown that Mixed Automata subsume PA. Tutorial [60] 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 [24] are an *interface model,* aiming to support specification, not programming. In addition to parallel com-

position, 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 rather simple; since choices remain local and statically defined, it is easy to turn them into probabilistic choices. For event structures with confusion, however, this is no longer the case: 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 cell,* 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 [18, 19], 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 [19], a link of such augmented nets is established with Bayesian networks, thus providing a result similar to ours in Section 2.2. Finally, [1, 2, 3] study trace monoids by equipping them with probabilities derived from local specifications, using analytic combinatoric techniques. As far as we know, this is the only approach supporting true concurrency with probabilistic choice and parallel composition, for infinite traces. Concurrency makes everything definitely harder.

6 Conclusion

We developed the model of Mixed (Probabilistic-Nondeterministic) Automata 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. It would make sense to develop an *interface theory* having Mixed Automata as models, along the lines of Abstract Probabilistic Automata [24]. 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 the notion of *consistency* for Mixed Systems. In the last appendix of [14], we give 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, extensively studied in [39]. 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 weaknesses and suggesting important bibliographical items while commenting the original version.

References

[1] Samy Abbes. Markov two-components processes. *Logical Methods in Computer Science*, 9(2), 2013.

[2] Samy Abbes. Synchronization of bernoulli sequences on shared letters. *Inf. Comput.*, 255:1–26, 2017.

[3] Samy Abbes. Markovian dynamics of concurrent systems. *Discrete Event Dynamic Systems*, 29(4):527–566, 2019.

[4] Samy Abbes and Albert Benveniste. Branching cells as local states for event structures and nets: Probabilistic applications. In Vladimiro Sassone, editor, *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*, volume 3441 of *Lecture Notes in Computer Science*, pages 95–109. Springer, 2005.

[5] Samy Abbes and Albert Benveniste. True-concurrency probabilistic models: Branching cells and distributed probabilities for event structures. *Inf. Comput.*, 204(2):231–274, 2006.

[6] Samy Abbes and Albert Benveniste. True-concurrency probabilistic models: Markov nets and a law of large numbers. *Theor. Comput. Sci.*, 390(2-3):129–170, 2008.

[7] Samy Abbes and Albert Benveniste. Concurrency, sigma-algebras, and probabilistic fairness. In Luca de Alfaro, editor, *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, volume 5504 of *Lecture Notes in Computer Science*, pages 380–394. Springer, 2009.

[8] Gilles Barthe, Thomas Espitau, Benjamin Grégoire, Justin Hsu, Léo Stefanesco, and Pierre-Yves Strub. Relational reasoning via probabilistic coupling. In Martin Davis, Ansgar Fehnker, Annabelle McIver, and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning - 20th International Conference, LPAR-20 2015, Suva, Fiji, November 24-28, 2015, Proceedings*, volume 9450 of *Lecture Notes in Computer Science*, pages 387–401. Springer, 2015.

[9] G. Baudart, L. Mandel, E. Atkinson, B. Sherman, M. Pouzet, and M. Carbin. Reactive probabilistic programming. In *Conference on Programming Language Design and Implementation (PLDI'20)*, London, UK, June 2020. To appear.

[10] Richard Bellman. A markovian decision process. *Journal of Mathematics and Mechanics*, 6(5):679–684, 1957.

[11] Albert Benveniste, Timothy Bourke, Benoît Caillaud, Jean-Louis Colaço, Cédric Pasteur, and Marc Pouzet. Building a hybrid systems modeler on synchronous languages principles. *Proceedings of the IEEE*, 106(9):1568–1592, 2018.

[12] Albert Benveniste, Paul Caspi, Stephen A. Edwards, Nicolas Halbwachs, Paul Le Guernic, and Robert de Simone. The synchronous languages 12 years later. *Proceedings of the IEEE*, 91(1):64–83, 2003.

[13] Albert Benveniste, Bernard C. Levy, Eric Fabre, and Paul Le Guernic. A calculus of stochastic systems for the specification, simulation, and hidden state estimation of mixed stochastic/nonstochastic systems. *Theor. Comput. Sci.*, 152(2):171–217, 1995.

[14] Albert Benveniste and Jean-Baptiste Raclet. Mixed Nondeterministic-Probabilistic Automata: Blending graphical probabilistic models with nondeterminism, 2022. Report to appear.

[15] David Blackwell. On a class of probability spaces. In *Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability, Volume 2: Contributions to Probability Theory*, pages 1–6, Berkeley, Calif., 1956. University of California Press.

[16] Johannes Borgström, Andrew D. Gordon, Michael Greenberg, James Margetson, and Jurgen Van Gael. Measure transformer semantics for bayesian machine learning. In *ESOP*, volume 6602 of *Lecture Notes in Computer Science*, pages 77–96. Springer, 2011.

[17] Johannes Borgström, Ugo Dal Lago, Andrew D. Gordon, and Marcin Szymczak. A lambda-calculus foundation for universal probabilistic programming. In *ICFP*, pages 33–46. ACM, 2016.

[18] Roberto Bruni, Hernán C. Melgratti, and Ugo Montanari. Concurrency and probability: Removing confusion, compositionally. *Log. Methods Comput. Sci.*, 15(4), 2019.

[19] Roberto Bruni, Hernán C. Melgratti, and Ugo Montanari. Bayesian network semantics for petri nets. *Theor. Comput. Sci.*, 807:95–113, 2020.

[20] Bob Carpenter, Andrew Gelman, Matthew Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Marcus Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell. Stan: A Probabilistic Programming Language. *Journal of Statistical Software, Articles*, 76(1):1–32, 2017.

[21] Krishnendu Chatterjee, Hongfei Fu, Petr Novotný, and Rouzbeh Hasheminezhad. Algorithmic analysis of qualitative and quantitative termination problems for affine probabilistic programs. *ACM Trans. Program. Lang. Syst.*, 40(2):7:1–7:45, 2018.

[22] Fredrik Dahlqvist and Dexter Kozen. Semantics of higher-order probabilistic programs with conditioning. *Proc. ACM Program. Lang.*, 4(POPL):57:1–57:29, 2020.

[23] Pedro R. D'Argenio, Pedro Sánchez Terraf, and Nicolás Wolovick. Bisimulations for nondeterministic labelled markov processes. *Math. Struct. Comput. Sci.*, 22(1):43–68, 2012.

[24] Benoît Delahaye, Joost-Pieter Katoen, Kim G. Larsen, Axel Legay, Mikkel L. Pedersen, Falak Sher, and Andrzej Wasowski. Abstract probabilistic automata. *Inf. Comput.*, 232:66–116, 2013.

[25] C. Dellacherie and PA. Meyer. *Probabilities and potentials.* North-Holland Mathematics Studies, North-Holland, Amsterdam, 1978.

[26] A. P. Dempster. Upper and lower probabilities induced by a multivalued mapping. *Ann. Math. Statist.*, 38(2):325–339, 04 1967.

[27] A. P. Dempster. A generalization of bayesian inference. *Journal of the Royal Statistical Society. Series B (Methodological)*, 30(2):205–247, 1968.

[28] Jerry den Hartog and Erik P. de Vink. Verifying probabilistic programs using a hoare like logic. *Int. J. Found. Comput. Sci.*, 13(3):315–340, 2002.

[29] Ernst-Erich Doberkat and Pedro Sánchez Terraf. Stochastic non-determinism and effectivity functions. *J. Log. Comput.*, 27(1):357–394, 2017.

[30] Norm Ferns, Prakash Panangaden, and Doina Precup. Bisimulation metrics for continuous markov decision processes. *SIAM J. Comput.*, 40(6):1662–1714, 2011.

[31] Marcus Gerhold and Mariëlle Stoelinga. Model-based testing of probabilistic systems. *Formal Aspects Comput.*, 30(1):77–106, 2018.

[32] Robert Givan, Thomas L. Dean, and Matthew Greig. Equivalence notions and model minimization in markov decision processes. *Artif. Intell.*, 147(1-2):163–223, 2003.

[33] Noah D. Goodman, Vikash K. Mansinghka, Daniel M. Roy, Keith Bonawitz, and Joshua B. Tenenbaum. Church: a language for generative models. *CoRR*, abs/1206.3255, 2012.

[34] Noah D Goodman and Andreas Stuhlmüller. The Design and Implementation of Probabilistic Programming Languages. http://dippl.org, 2014. Accessed: 2021-4-20.

[35] Andrew D. Gordon, Thomas A. Henzinger, Aditya V. Nori, and Sriram K. Rajamani. Probabilistic programming. In James D. Herbsleb and Matthew B. Dwyer, editors, *Proceedings of the on Future of Software Engineering, FOSE 2014, Hyderabad, India, May 31 - June 7, 2014*, pages 167–181. ACM, 2014.

[36] Vineet Gupta, Radha Jagadeesan, and Prakash Panangaden. Stochastic processes as concurrent constraint programs. In *POPL*, pages 189–202. ACM, 1999.

[37] P.R. Halmos. *Measure Theory.* Graduate Texts in Mathematics. Springer New York, 1976.

[38] Justin Hsu. Probabilistic couplings for probabilistic reasoning. *CoRR*, abs/1710.09951, 2017.

[39] Chung-Kil Hur, Aditya V. Nori, Sriram K. Rajamani, and Selva Samuel. A provably correct sampler for probabilistic programs. In Prahladh Harsha and G. Ramalingam, editors, *35th IARCS Annual Conference on Foundation of Software Technology and Theoretical Computer Science, FSTTCS 2015, December 16-18, 2015, Bangalore, India*, volume 45 of *LIPIcs*, pages 475–488. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2015.

[40] C. Jones and Gordon D. Plotkin. A probabilistic powerdomain of evaluations. In *LICS*, pages 186–195. IEEE Computer Society, 1989.

[41] G. David Forney Jr. Maximum-likelihood sequence estimation of digital sequences in the presence of intersymbol interference. *IEEE Trans. Information Theory*, 18(3):363–378, 1972.

[42] Joost-Pieter Katoen, Friedrich Gretz, Nils Jansen, Benjamin Lucien Kaminski, and Federico Olmedo. Understanding probabilistic programs. In Roland Meyer, André Platzer, and Heike Wehrheim, editors, *Correct System Design - Symposium in Honor of Ernst-Rüdiger Olderog on the Occasion of His 60th Birthday, Oldenburg, Germany, September 8-9, 2015. Proceedings*, volume 9360 of *Lecture Notes in Computer Science*, pages 15–32. Springer, 2015.

[43] Ross Kindermann and Laurie Snell. *Markov random fields and their applications*, volume 1. American Mathematical Society, 1980.

[44] Dexter Kozen. Semantics of probabilistic programs. *J. Comput. Syst. Sci.*, 22(3):328–350, 1981.

[45] E. L. Lehmann and Joseph P. Romano. *Testing statistical hypotheses.* Springer Texts in Statistics. Springer, New York, third edition, 2005.

[46] H. . Loeliger. An introduction to factor graphs. *IEEE Signal Processing Magazine*, 21(1):28–41, 2004.

[47] David Lunn, David Spiegelhalter, Andrew Thomas, and Nicky Best. The BUGS project: Evolution, critique and future directions. *Statistics in Medicine*, 28(25):3049–3067, 2009.

[48] Nancy A. Lynch, Roberto Segala, and Frits W. Vaandrager. Compositionality for probabilistic automata. In *Proc. of the 14th International Conference on Concurreny Theory (CONCUR'03)*, volume 2761 of *Lecture Notes in Computer Science*, pages 204–222. Springer, 2003.

[49] Annabelle McIver and Carroll Morgan. *Abstraction, Refinement and Proof for Probabilistic Systems.* Monographs in Computer Science. Springer, 2005.

[50] Annabelle McIver and Carroll Morgan. Correctness by construction for probabilistic programs. In Tiziana Margaria and Bernhard Steffen, editors, *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*, volume 12476 of *Lecture Notes in Computer Science*, pages 216–239. Springer, 2020.

[51] Federico Olmedo, Friedrich Gretz, Nils Jansen, Benjamin Lucien Kaminski, Joost-Pieter Katoen, and Annabelle McIver. Conditioning in probabilistic programming. *ACM Trans. Program. Lang. Syst.*, 40(1):4:1–4:50, 2018.

[52] Judea Pearl. Fusion, propagation, and structuring in belief networks. *Artif. Intell.*, 29(3):241–288, 1986.

[53] Judea Pearl. *Causality*. Cambridge University Press, Cambridge, UK, 2 edition, 2009.

[54] Martyn Plummer. 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. K Hornik, F Leisch, A Zeileis (eds.).

[55] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[56] L. R. Rabiner and B. H. Juang. An introduction to hidden markov models. *IEEE ASSp Magazine*, 1986.

[57] Roberto Segala. Probability and nondeterminism in operational models of concurrency. In *Proc. of the 17th International Conference on Concurrency Theory (CONCUR'06)*, volume 4137 of *Lecture Notes in Computer Science*, pages 64–78. Springer, 2006.

[58] Roberto Segala and Nancy A. Lynch. Probabilistic simulations for probabilistic processes. In *Proc. of the 5th International Conference on Concurrency Theory (CONCUR'94)*, volume 836 of *Lecture Notes in Computer Science*, pages 481–496. Springer, 1994.

[59] Glenn Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, Princeton, 1976.

[60] Ana Sokolova and Erik P. de Vink. Probabilistic automata: System types, parallel composition and comparison. In Christel Baier, Boudewijn R. Haverkort, Holger Hermanns, Joost-Pieter Katoen, and Markus Siegle, editors, *Validation of Stochastic Systems - A Guide to Current Research*, volume 2925 of *Lecture Notes in Computer Science*, pages 1–43. Springer, 2004.

[61] Sam Staton, Hongseok Yang, Frank D. Wood, Chris Heunen, and Ohad Kammar. Semantics for probabilistic programming: higher-order functions, continuous distributions, and soft constraints. In Martin Grohe, Eric Koskinen, and Natarajan Shankar, editors, *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '16, New York, NY, USA, July 5-8, 2016*, pages 525–534. ACM, 2016.

[62] Daniel W. Stroock and S. R. S. Varadhan. 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*, pages 333–359, Berkeley, Calif., 1972. University of California Press.

[63] Regina Tix, Klaus Keimel, and Gordon Plotkin. Semantic domains for combining probability and non-determinism. *Electr. Notes Theor. Comput. Sci.*, 222:3–99, 01 2009.

[64] David Tolpin, Yuan Zhou, Tom Rainforth, and Hongseok Yang. Probabilistic programs with stochastic conditioning. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 10312–10323. PMLR, 2021.

[65] Jan-Willem van de Meent, Brooks Paige, Hongseok Yang, and Frank Wood. An introduction to probabilistic programming, 2018.

[66] Di Wang, Jan Hoffmann, and Thomas W. Reps. A denotational semantics for low-level probabilistic programs with nondeterminism. In Barbara König, editor, *Proceedings of the Thirty-Fifth Conference on the Mathematical Foundations of Programming Semantics, MFPS 2019, London, UK, June 4-7, 2019*, volume 347 of *Electronic Notes in Theoretical Computer Science*, pages 303–324. Elsevier, 2019.

In this supplementary material, we first collect all the missing proofs. Then, with reference to Footnote 2, we include a short discussion of how to extend our model by relaxing the restriction that probability spaces should all be at most denumerable.

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 [49, 50]. This line of work addresses probabilistic extensions of Hoare logic for imperative programs, focusing on evaluating the probability of weakest preconditions of properties. 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 chosed to base our comparison on a different work in the same direction: [21], which provides the most extensive developement on *demonic/angelic* blending of probability and nondeterminism in the language Apps. We do not claim to cover all aspects of Apps, since the focus of this reference is on the checking of 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 [21], 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

$$P : x \text{ gets increased by one by performing } Q_3\,. \tag{44}$$

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};

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

Q_1

Q_2

Q_3

Verbatim from [21]: 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 [21], call Q_3 the body of the `while` loop of this example: `while` $x \geq 0$ `do` Q_3.

Figure 6: Example of Fig. 2 of [21]

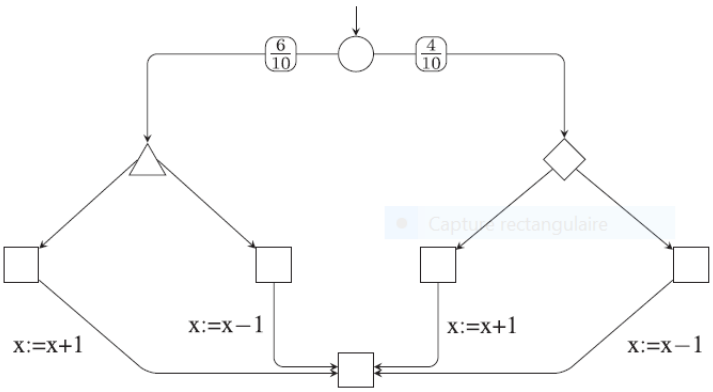

Figure 7: Semantics: SGS (Stochastic Game Structure) of Q_3, Fig. 6 of [21]. The execution begins with the probabilistic choice. The left branch (corresponding to Q_1) is selected according to demonic nondeterminism figured by a triangle, and the right branch (corresponding to Q_2) is selected according to angelic nondeterminism, figured by a diamond.

- It remains to define relation C involving ω, x, x'. To mimic Fig. 7, we would like to write something like

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

Unfortunately, **angelic/demonic choice** are not concepts of our Mixed Systems model following Definition 1. With regard to 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 both alternatives for different parts of the system.

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

$$S = (\Omega, \pi, X, C)\,, \tag{45}$$

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

$$\omega C x' \text{ iff } \begin{cases} \omega = Q_1 \quad \wedge \quad \textbf{angel } x' \in \{x{-}1, x{+}1\} \\ \text{or} \\ \omega = Q_2 \quad \wedge \quad \textbf{demon } x' \in \{x{-}1, x{+}1\} \end{cases} \tag{46}$$

This definition for C is informal, since keywords **demon** and **angel** have no mathematical meaning by themselves. We will give a semantics to (46) 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:

$$\begin{array}{rcl} \pi^*(\neg P) & =_{\text{def}} & \pi^{\mathsf{c}}\left(\{\omega \mid \omega{=}Q_1 \wedge \exists x'{\in}\{x{-}1, x{+}1\} : \neg P\}\right) \\ & & +\pi^{\mathsf{c}}\left(\{\omega \mid \omega{=}Q_2 \wedge \forall x'{\in}\{x{-}1, x{+}1\} : \neg P\}\right) \end{array} \tag{47}$$

In this formula, we give a semantics to **angel** in (46) by using the existential quantifier, i.e., we use the outer probability to evaluate the corresponding state predicate; we give a semantics to **demon** in (46) 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 (46), for both choices $\omega = Q_1$ and $\omega = Q_2$, related values for state x' exist. Formula (47) finally yields $\pi^*(\neg P) = 0.6$.

The above coding applies only to a restricted class of relations C. In formula (47), we exploited the fact that, in relation C defined by (46), a partition of Ω is performed first (probabilistic choice), and then, each branch of this choice involves a pure state predicate, independent from ω.

Here follow 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 [21]. Since Mixed Systems do not support recursion, we consider only the subclass of SGS that are DAGs. 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 (45,46,47), this yields the whole SGS. For each such subgraph, a coding similar to (45,46,47) can be given. The partially ordered execution of the whole SGS is then mapped to a Bayesian network following Definition 9, and 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, first, it applies only to a restricted class of relations C, and, second, we believe it to be incompatible with having a parallel composition.

A.2 Proof of Lemma 7

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

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

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.

A.3 Proof of Theorem 13

Proof: We will repeatedly use notation (27). 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:

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

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 (22). 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 (9), (21) and notation (22), 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

$$\pi_Y^{\mathsf{c}}\big((C_Y)_{q_{*Y}}\big) \tag{48}$$

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

$$\pi^{\mathsf{c}}\big(C_{q_*} \mid (C_Y)_{q_{*Y}}\big)\,, \tag{49}$$

which is well defined since $\pi^{\mathsf{c}}_Y((C_Y)_{q_{*Y}}) > 0$.

3. By (26), the probabilistic score of q_* is equal to the product of the two scores (48) and (49):

$$\pi^{\mathsf{c}}\big(C_{q_*} \mid (C_Y)_{q_{*Y}}\big)\ \pi^{\mathsf{c}}_Y\big((C_Y)_{q_{*Y}}\big) \;=\; \pi^{\mathsf{c}}\big(C_{q_*} \cap (C_Y)_{q_{*Y}}\big) \;=\; \pi^{\mathsf{c}}\big(C_{q_*}\big)\,,$$

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.

A.4 Proof of Theorem 14

As a prerequisite, we need the following result:

Lemma 23 *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 $\mathbf{Pr}_{X_1 \cup Y}(C_1 \wedge C_2) = C_1 \wedge \mathbf{Pr}_{X_1 \cup Y}(C_2) = C_1 \wedge \mathbf{Pr}_Y(C_2)$. □

The proof of Theorem 14 relies on the following lemma, which is a corollary of Bayes formula. This lemma provides the basic reasoning step of message passing algorithms:

Lemma 24 *Let S_1, S_2, and Y be as in Lemma 23. Then:*

$$S_1 \parallel S_2 \quad \equiv_P \quad \big(S_1 \parallel \mathbf{Margin}_Y(S_2)\big);\, \mathbf{Cond}_Y(S_2)\ . \tag{50}$$

Proof: For proving formula (50), we first apply Theorem 13 with S replaced by $S_1 \parallel S_2$, which yields: $S_1 \parallel S_2 \quad \equiv_P \quad \mathbf{Margin}_{X_1 \cup Y}(S_1 \parallel S_2)\,;\mathbf{Cond}_Y(S_1 \parallel S_2)$. Then, by Lemma 23, $\mathbf{Margin}_{X_1 \cup Y}(S_1 \parallel S_2) \equiv S_1 \parallel \mathbf{Margin}_Y(S_2)$ 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 ";". □

Having proved this lemma, the proof of Theorem 14 reproduces exactly the reasoning steps establishing the message passing algorithm mapping factor graphs to Bayesian Networks in the classical setting [46]; thus we only sketch here the argument of the proof. *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

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

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 (50) of Lemma 24, reorganize $\mathcal{S}$ by rewriting

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

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

B Proofs Regarding Mixed Automata

B.1 Proof of Lemma 18

Proof: The result is immediate if both S_1 and S_1' are compressed, see Definition 2. 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

$$\pi_1(\omega_1) = \textstyle\sum_{\omega_1'\in\omega_1}\pi_1'(\omega_1') \tag{51}$$

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 (14) 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:

$$\begin{array}{rl} \forall\omega_1': & \pi_1'(\omega_1') = \sum_{\omega_2} w'(\omega_1',\omega_2) \\ \forall\omega_2: & \pi_2(\omega_2) = \sum_{\omega_1'} w'(\omega_1',\omega_2) \\ \forall(\omega_1',\omega_2;q_1): & \left[\begin{array}{c} w'(\omega_1',\omega_2)>0 \\ \omega_1'\,C_1'\,q_1 \end{array}\right] \Rightarrow \exists q_2 : \left[\begin{array}{c} \omega_2\,C_2\,q_2 \\ q_1\,\rho\,q_2 \end{array}\right] \end{array} \tag{52}$$

Focus first on the first two lines of (52). The following calculation shows that

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

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 (52):

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

We move to the third line of (52). 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.

B.2 Proof of Lemma 20

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.

C Proofs Regarding the comparison with Probabilistic Automata

C.1 Proof of Theorem 21 regarding Simple Probabilistic Automata

C.1.1 Statement 1 of Theorem 21: 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

$$M_P \quad = \quad (\Sigma, \{\xi\}, q_0, \rightarrow_P), \tag{53}$$

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 (53).

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 [57], corresponds to our Definition 17, 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 19, when restricted to purely probabilistic mixed systems. This proves the part of Theorem 21 regarding simulation.

Mapping parallel composition: We move to parallel composition, for which the reader is referred to [48], 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

$$(q_1, q_2) \xrightarrow{\alpha} \pi_1 \times \pi_2 \quad \text{iff} \quad q_i \xrightarrow{\alpha}_i \pi_i \text{ for } i = 1, 2 \tag{54}$$

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:

$$(q_1, q_2) \xrightarrow{\alpha}_{12} S_1 \parallel S_2 \quad \text{iff} \quad q_i \xrightarrow{\alpha}_i S_i \text{ for } i = 1, 2 \tag{55}$$

We thus need to show that

$$M_P \text{ and } M \text{ are simulation equivalent.} \tag{56}$$

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 (54) and (55) 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 (55). 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 (56) follows.

C.1.2 Statement 2 of Theorem 21: 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:

$$\rightarrow_{P_M} \quad = \quad \{(p, \alpha, \mu) \mid \exists S : (p, \alpha, S) \in \rightarrow_M \text{ and } \mu \in \mathbf{P}_S\} \tag{57}$$

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 (57), 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 $Q \times Q'$:

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

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 25 *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 8 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

$$\mathbf{x} : \Omega_1 \uplus \Omega_2 \to Q_x, \text{ such that } \begin{cases} \mathbf{x}(\omega_1) = q \text{ if } \omega_1 = (q_1, q) \\ \mathbf{x}(\omega_2) = q' \text{ if } \omega_2 = (q', q_2) \end{cases} \tag{58}$$

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 (58); 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 $M{=}(\{\alpha\}, X, q_0, \rightarrow)$, 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 $\rightarrow$ maps, through action α, any state $p{\in}Q$ to the above system $S(p)$. Similarly, we consider the two Mixed Automata $M_i = (\{\alpha\}, X_i, q_{i,0}, \rightarrow_i)$, $i{=}1, 2$, where X_i is as above, $q_{i,0}$ is the projection of q_0 on Q_i, and $\rightarrow_i$ 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 $M_1 \parallel M_2$.

C.2 Proof of Theorem 22 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

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

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

D Extending Mixed Systems to continuous probabilities

In this appendix, we indicate how to relax the restriction that the considered probability spaces should all be discrete and we discuss technical difficulties. A recommended reference on probability theory is [25]. The reader is invited to compare the following writing with the corresponding material of Section 2. We begin with some notations and prerequisites.

D.1 Notations and prerequisites on probability theory

For P and Q two sets, $P \times Q$ their product, and $A \subseteq P \times Q$, we denote by $\mathbf{Pr}_P(A)$ the projection of A over P.

Probability spaces: $(\Omega, \mathcal{F}, \pi)$ shall generically denote a probability space, i.e., Ω is a set, $\mathcal{F}$ is a σ-algebra over Ω (i.e., a subset of 2^Ω, containing $\emptyset$ and stable under complement, and countable unions and intersections), and π is a probability (i.e., a countably additive function, from σ-algebra $\mathcal{F}$ to $[0,1]$, such that $\pi(\emptyset) = 0$ and $\pi(\Omega) = 1$). Let $p : (\Omega, \mathcal{F}) \mapsto \{0,1\}$ be a measurable predicate, say that p *holds almost everywhere* if $\pi\{\omega \mid p(\omega) = 1\} = 1$. For a measurable function $f : (\Omega, \mathcal{F}) \mapsto (\mathbb{R}_+, \mathcal{L})$, where $\mathcal{L}$ is the Borel σ-algebra over $\mathbb{R}_+$, we write

$$\mathbb{E}(f) \quad =_{\text{def}} \quad \int f(\omega)\pi(d\omega)\,.$$

For $(\Omega_i, \mathcal{F}_i, \pi_i)_{i=1,2}$ two probability spaces, $\mathcal{F}_1 \times \mathcal{F}_2$ is defined as the smallest σ-algebra over $\Omega_1 \times \Omega_2$ making the two projections measurable, and $\pi_1 \times \pi_2$ shall denote the cartesian product of the two probabilities, characterized by $(\pi_1 \times \pi_2)(A_1 \times A_2) = \pi_1(A_1)\pi_2(A_2)$, where $A_i \in \mathcal{F}_i$. Infinite products of probabilities $\pi = \prod_{i\in I} \pi_i$ with arbitrary index set I can even be defined; they are characterized by the equalities $\pi\left(\prod_{i\in I} A_i\right) = \prod_{i\in I} \pi_i(A_i)$, where all but a finite number of A_i are equal to Ω_i.

Conditional expectations and conditional probabilities: For $\mathcal{G} \subseteq \mathcal{F}$ a sub-σ-algebra of $\mathcal{F}$, and $X : (\Omega, \mathcal{F}) \mapsto \mathbb{R}_+$, measurable, there exists $Y : (\Omega, \mathcal{G}) \mapsto \mathbb{R}_+$, measurable, such that $\mathbb{E}(X \times Z) = \mathbb{E}(Y \times Z)$ for any measurable $Z : (\Omega, \mathcal{G}) \mapsto \mathbb{R}_+$. Y satisfying the above properties is almost surely unique: $\pi(Y' \neq Y) = 0$ for any two such random variables. Y is called the

$$\textit{conditional expectation of } X \textit{ given } \mathcal{G}\text{, written } \mathbb{E}(X \mid \mathcal{G}). \tag{59}$$

For $A \in \mathcal{F}$, let $\mathbf{1}_A$ denote the characteristic function of set A, which equals 1 on A and 0 elsewhere; then, we write

$$\pi(A \mid \mathcal{G}) \quad =_{\text{def}} \quad \mathbb{E}(\mathbf{1}_A \mid \mathcal{G}). \tag{60}$$

If $B \in \mathcal{F}$ satisfies $\pi(B) > 0$ and $\mathcal{F}_B = \{\emptyset, \Omega, B, \Omega\backslash B\}$ is the smallest σ-algebra containing set B, then, the conditional expectation $f =_{\text{def}} \pi(A \mid \mathcal{F}_B)$ is such that $f(\omega) = \pi(A \mid B) = \frac{\pi(A\cap B)}{\pi(B)}$ for almost every $\omega \in B$. To show this, we form

$$\mathbb{E}(\pi(A \mid \mathcal{G}) \times \mathbf{1}_B) = \mathbb{E}\left(\tfrac{\pi(A\cap B)}{\pi(B)} \times \mathbf{1}_B\right) = \tfrac{\pi(A\cap B)}{\pi(B)} \times \underbrace{\mathbb{E}\left(\mathbf{1}_B\right)}_{=\pi(B)} = \pi(A\cap B)$$

with a similar result for $\mathbf{1}_{\Omega-B}$, showing that the characterization of the conditional expectation is satisfied. This establishes the link between conditional expectation and conditional probability in its elementary setting.

For $\mathcal{G}_2, \mathcal{G}_1 \subseteq \mathcal{F}$ two sub-σ-algebras, and $X : (\Omega, \mathcal{F}) \mapsto \mathbb{R}_+$, measurable, we write $\mathbb{E}(X \mid \mathcal{G}_1 \mid \mathcal{G}_2) =_{\text{def}} \mathbb{E}(\mathbb{E}(X \mid \mathcal{G}_1) \mid \mathcal{G}_2)$. Let $B \in \mathcal{F}$ and $\mathcal{G} \subseteq \mathcal{F}$, and let $\mathcal{F}_B$ be the smallest σ-algebra containing the set B; we write

$$\pi(A \mid \mathcal{F}_B \mid \mathcal{G}) \quad =_{\text{def}} \quad \mathbb{E}(\,\pi(A \mid \mathcal{F}_B) \mid \mathcal{G}) = \mathbb{E}(\,\mathbb{E}(\mathbf{1}_A \mid \mathcal{F}_B) \mid \mathcal{G})\,. \tag{61}$$

Disintegration: Consider $(\Omega, \mathcal{F}, \pi)$ and $\mathcal{G} \subseteq \mathcal{F}$ as before. So far we have defined $\pi(A \mid \mathcal{G})$ as a $\mathcal{G}$-measurable random variable, for a given $A \in \mathcal{F}$. Can we take it as a *transition probability* $P(\omega, A)$, i.e., a map such that $A \mapsto P(\omega, A)$ is a probability for ω fixed, and $\omega \mapsto P(\omega, A)$ is $\mathcal{G}$-measurable for A fixed? Here is the formalization:

Definition 26 ([25, 37]) *Call* disintegration[4] *$\pi(A \mid \mathcal{G})$, where A ranges over $\mathcal{F}$, a map $(\omega, A) \mapsto P(A \mid \omega)$ from $\Omega \times \mathcal{F}$ to $[0, 1]$ such that:*

1. *For A fixed, $\omega \mapsto P(A \mid \omega)$ is $\mathcal{G}$-measurable, and $P(A \mid \cdot)$ is a version of the conditional expectation $\pi(A \mid \mathcal{G})$;*
2. *For ω fixed, $A \mapsto P(A \mid \omega)$ is a probability.*

If $P(A \mid \omega)$ and $P'(A \mid \omega)$ are two such regular versions, then, probabilities $P(\cdot \mid \omega)$ and $P'(\cdot \mid \omega)$ must be equal outside a set of ω of π-probability zero.

Existence of a disintegration: The existence of a disintegration is not always guaranteed. It is obvious for discrete probability spaces. It is not true in general, however, see, e.g., [15]. Failure to exist typically occurs when working with measurable spaces completed with subsets of zero probability sets. The existence of a a disintegration is only guaranteed under specific topological properties for the underlying set. Jirina Theorem is an example of broad sufficient topological conditions for the existence of regular versions for conditional expectations, see [25, 37]. The Blackwell spaces, however, are adequate tools for this, so we introduce them next. For the following material, the reader is referred to [15].

If Ω is a metric space, the smallest σ-algebra containing all open sets of Ω is its *Borel σ-algebra,* denoted by $\mathcal{B}$. Borel σ-algebra $\mathcal{B}$ is called *separable* if there is a sequence $B_n \in \mathcal{B}$ such that $\mathcal{B}$ is the smallest Borel σ-algebra containing all B_n. In particular, if Ω is a separable metric space, its Borel σ-algebra is separable. The *atoms* of $\mathcal{B}$ are the sets $B \in \mathcal{B}$ such that no proper nonempty subset of it belongs to $\mathcal{B}$. Any two nonidentical atoms are disjoint and every Borel set is a union of atoms.

A metric space A will be called *analytic* if A is the continuous image of the set of irrational numbers. The following properties hold, showing which cases are covered by this notion:

1. If A_n is a sequence of analytic sets in a metric space Ω, then $\bigcup_n A_n$, $\bigcap_n A_n$ if nonempty, the product space $A_1 \times A_2$ and the infinite product space $A_1 \times A_2 \times \cdots$, are analytic sets.
2. If A is analytic, so is every Borel subset of A.
3. Every Borel set of the Euclidean n-space is analytic.
4. If A, B are disjoint analytic subsets of a metric space Ω, there is a Borel set D of Ω such that $D \supset A$ and $D \cap B = \emptyset$.

[4] Depending on the authors, disintegration is also called *regular version of the conditional expectation.*

5. If f is a Borel-measurable mapping of an analytic set A into a separable metric space Q, then $f(A)$, the range of f, is an analytic set.

Pairs $(\Omega, \mathcal{B})$, where Ω is analytic and $\mathcal{B}$ is its Borel σ-algebra, are called *Blackwell spaces*.[5] The following two results are proved in [15]:

Theorem 27 *For $(\Omega, \mathcal{B}, \pi)$ a Blackwell space:*

1. *Two separable sub-σ-algebras of $\mathcal{B}$ with the same atoms are identical.*
2. *For π any probability on $(\Omega, \mathcal{B}, \pi)$ and $\mathcal{B}'$ any separable sub-σ-algebra of $\mathcal{B}$, there exists a disintegration for $\pi(A \mid \mathcal{B}')$.*

In the following we will assume (unless otherwise stated) that all considered measurable spaces are Blackwell, so that Theorem 27 can be applied.

D.2 Definition and basic properties

Relations: Upper case letters X, Y, Z shall denote finite sets of *variables*, and variables are denoted by corresponding lower case letters $x, y, z \ldots$ Let the domain of x be denoted by Q_x and be equipped with a σ-algebra $\mathcal{G}_x$; the domain of X is $Q_X =_{\text{def}} \prod_{x\in X} Q_x$, equipped with the product σ-algebra $\mathcal{G}_X = \prod_{x\in X} \mathcal{G}_x$. We will consider *equations* (also called *relations* or *constraints*): an equation on X identifies with its set of solutions, i.e., a measurable subset of Q_X; if $Y \subseteq X$, an equation on Y can be seen as an equation on X. We consider *systems of equations*, which are sets of equations implicitly composed via intersection.

Definition 28 (Mixed System) *A* Mixed System *is a tuple*

$$S \quad = \quad \big((\Omega, \mathcal{F}, \pi), (Q_x, \mathcal{G}_x)_{x\in X}, C\big), \tag{62}$$

where $(\Omega, \mathcal{F}, \pi)$ is a private probability space; $(Q_x, \mathcal{G}_x)_{x\in X}$ is a finite set of measurable state spaces with product $(Q, \mathcal{G}) =_{\text{def}} \prod_{x\in X}(Q_x, \mathcal{G}_x)$, and $C \in \mathcal{F}\times\mathcal{G}$ is a measurable relation over $\Omega\times Q$. In the sequel, we also write $\omega C q$ to mean $(\omega, q) \in C$, and we identify the set of variables X with the measurable state space $(Q_x, \mathcal{G}_x)_{x\in X}$ it defines, thus we write

$$S \quad = \quad \big((\Omega, \mathcal{F}, \pi), X, C\big), \tag{63}$$

for short instead of (62).

Defining the semantics of Mixed Systems in the general case requires some care, as the following example shows.

Example 9 [discussing consistency] Let X and Y be two real random variables with continuous joint distribution π. Formally, $\Omega = \mathbb{R}^2$, $\mathcal{F}$ is the Lebesgue σ-algebra over Ω, π is a continuous probability over $(\Omega, \mathcal{F})$ and X and Y are the first and second coordinates of $\mathbb{R}^2$. For y a given value for Y, consider $C = \{(\omega, y) \mid \omega \in \Omega\}$ completing the definition of Mixed System $S = \big((\Omega, \mathcal{F}, \pi), X, C\big)$. The intuition is that S models the conditional distribution of (X, Y) given that $Y = y$. We would like this to be a consistent system, despite $\pi(Y{=}y) = 0$. Thus, elementary Definition 1 for the operational semantics cannot be used since it would lead to considering system S as inconsistent.

[5] They are actually called "Lusin spaces" in [15], but the term "Backwell spaces" has been used since then in the literature to avoid the confusion with the hierarchy of Polish topological spaces.

For this case, the correction is easily guessed. The aim is that prior probability π should be replaced by the posterior conditional distribution $\pi(\cdot \mid y)$, rather a disintegration for it. This amounts to making y "variable" by considering a disintegration $\pi(A \mid \mathcal{F}_Y)$ according to Definition 26, where $\mathcal{F}_Y \subset \mathcal{F}$ is the σ-algebra generated by random variable Y. Recall that the existence of such a disintegration is subject to topological conditions, see the comment following Definition 26—such conditions are satisfied by this example. Then, we take $\pi^{\mathsf{c}} = \pi(\cdot \mid y)$ by taking the corresponding disintegration. □

How can we extend this to general Mixed Systems? Informally, how can we make relation C "variable"?

Definition 29 (consistency and sampling) *Mixed System S is called* consistent *if the following conditions hold:*

1. *There exists a sub-σ-algebra $\mathcal{H} \subseteq \mathcal{F}$ such that a disintegration $\pi(\cdot \mid \mathcal{H})$ exists; we denote by $\mathbf{a}$ a generic atom of $\mathcal{H}$, thus conditional probability $\pi(\cdot \mid \mathcal{H})$ becomes a function of atom $\mathbf{a}$, so we write it $\pi(\cdot \mid \mathbf{a})$;*
2. *There exists a measurable relation $\mathcal{C} \in \mathcal{F} \times \mathcal{G}$ such that*
   (a) *Relation C takes the form $C = \mathcal{C} \cap (\mathbf{a} \times Q)$ for some atom $\mathbf{a}$ of $\mathcal{H}$;*
   (b) *$\pi(\Omega^{\mathsf{c}} \mid \mathbf{a}) > 0$ where $\Omega^{\mathsf{c}} =_{\text{def}} \{\omega \mid \exists q : \omega \mathcal{C} q\}$.*

If S is consistent, define π^{c} by

$$\pi^{\mathsf{c}}(A) \quad =_{\text{def}} \quad \frac{\pi(A \cap \Omega^{\mathsf{c}} \mid \mathbf{a})}{\pi(\Omega^{\mathsf{c}} \mid \mathbf{a})} \tag{64}$$

The sampling *of S consists in:* (1) *drawing ω at random using π^{c}, and* (2) *nondeterministically selecting q such that $\omega C q$. This two-step procedure is denoted by $S \rightsquigarrow q$.*

The "variable embedding" of C is the relation $\mathcal{C}$, from which C is retrieved by selecting the atom w at step 2a.

Example 10 Consider the ReactiveBayes program "$S_1 \parallel S_2 \parallel S_3 \parallel S_4$" of the introduction. Now, the white noise model for `w` in `Noise` is truly Gaussian (or any other distribution on $\mathbb{R}$, possibly continuous). The prior probability of this system is

$$\text{prior proba} : \begin{cases} rf_n \sim \textbf{Bernoulli}(10^{-6}) \\ v_n \sim \mu \\ \text{by semantic convention, } rf \text{ and } w \text{ are independent} \end{cases} \tag{65}$$

and relation C is the following system of equations:

$$C \quad : \quad \begin{cases} \texttt{observe}\ u, y \\ x_0 = c_x\ ,\ v_0 = c_v\ ,\ f_0 = \texttt{F} \\ x_n = \varphi(u_n, x_{n-1}) \\ y_n = \text{if } f_n \text{ then } \psi(x_n, v_n) \text{ else } x_n \\ f_n = (rf_n \text{ or } f_{n-1}) \text{ and not } bk_n \end{cases} \tag{66}$$

The following observation is the key to handle the model (65,66): if we forget for a while the first constraint **observe** y in (66), then the resulting dynamical system can be seen as an input/output system with inputs u, v, rf, of which u is a measured input, whereas v, rf are random inputs:

there is nothing unusual. In this i/o system, the prior probability is not subject to any constraint, hence the posterior probability equals the prior.

The difficulty comes with the consideration of the output constraint `observe` y. This suggests taking for the instrumental σ-algebra $\mathcal{H}$ the σ-algebra generated by y. Accordingly, we partition (66) as

$$C \;:\; \begin{cases} & \textbf{observe } y & \text{defining } \mathcal{H}, \text{ the } \sigma\text{-algebra generated by } y, \\ & & \text{whose atom is represented by a value for } y. \\ y_n \;=\; & f(v_n, rf_n) & \text{defining } \mathcal{C} \text{ and consistency set } \Omega^{\mathsf{c}}, \text{ equal to } \Omega \end{cases} \tag{67}$$

where f is the function resulting from computing y from the pair (w, rf) by using system of equations (66) in which the first equation has been deleted (other variables are also computed). This defines the auxiliary relation $\mathcal{C}$ and we have $\Omega^{\mathsf{c}} = \Omega$. The σ-algebra $\mathcal{F}$ is generated by the pair (w, rf) of random variables, and $\mathcal{H}$ is the σ-algebra generated by $y =_{\text{def}} f(w, rf)$. Atoms of $\mathcal{H}$ consist of any reachable value for y. Then, $\pi(\cdot \mid \mathcal{H}) = \pi(\cdot \mid y)$ is the conditional distribution of the pair (w, rf) given a reachable value for y, and $\Omega^{\mathsf{c}} = \Omega$, showing that the considered system is consistent. □

As a side result, the discussion of this example suggests how sampling can be performed in practice for ReactiveBayes programs involving continuous distributions.

RESEARCH CENTRE
RENNES – BRETAGNE ATLANTIQUE

Campus universitaire de Beaulieu
35042 Rennes Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399