Bayesian updates from determinisation

Manuel Baltieri
Araya Inc., Japan*
manuel_baltieri@araya.org

Nathaniel Virgo
University of Hertfordshire, UK*
n.virgo@herts.ac.uk

The powerset construction is the classical determinisation procedure for nondeterministic finite automata. In the coalgebraic setting, this construction has been generalised to structured coalgebras, coalgebras parametrised by an endofunctor, a monad, and a compatible algebra structure. For stochastic Moore machines over the distribution monad, a type of structured coalgebra, the resulting determinised machine has a state space of distributions on states and transitions given by the pushforwards of these distributions along the appropriate maps. It then also induces a semantics assigning to each finite input word a distribution on the current output. This semantics is appropriate when only the current output matters, but it is too coarse for settings in which intermediate observations must also be taken into account. These settings are typical for agents solving POMDPs in control theory and reinforcement learning. In these contexts, agents need update state distributions by conditioning on all realised observations, not just the final one, so to better plan for the future. Inspired by computational mechanics and its treatment of unifilar models for given hidden Markov models tackling a similar problem, a categorical construction with this feature has been proposed under the name of *unifilarisation*. Starting from a stochastic Mealy machine, it produces a machine whose states are priors over the original state space and whose transitions are given by Bayesian filtering. In this paper, we show that unifilarisation is nevertheless an instance of coalgebraic determinisation. More precisely, we work with Mealy machines over monads equipped with extra structure generalising the notion of the support of a distribution. We show that in this setting, the corresponding unifilar machines arise from the general determinisation procedure. In the stochastic case, the induced transition takes a prior-weighted joint of next output and next state and decomposes it into an output marginal together with posterior next-state distributions. We then compare the resulting final coalgebra semantics with the Moore-style one. Instead of assigning only a distribution on current outputs to each finite input word, it yields causal stochastic behaviours, that is, families mapping input words to distributions on output words compatible with Bayesian filtering updates.

1 Introduction

Determinisation is classically introduced in automata theory as the move from nondeterministic finite automata to deterministic finite automata via the subset (or powerset) construction [RS59]: given an nondeterministic finite automata with state set X, the determinised machine has state set $\mathcal{P}_{\text{fin}}(X)$, with a transition function mapping a subset $S \subseteq X$ and an input symbol a to the set of all a-successors of states in S, i.e. all the states that can be reached from S after consuming an input symbol a. This construction yields an equivalent deterministic automaton that accepts the same language of finite words.

This construction has previously been generalised to a coalgebraic setup, specifically for (structured) coalgebras $(X, f : X \to \mathrm{FT}(X))$ that can be *determinised* to coalgebras $\big(\mathrm{T}(X), f^\sharp : \mathrm{T}(X) \to \mathrm{FT}(X)\big)$ [SBBR10, SBBR13, JSS15, BSS21]. This generalised construction is achieved by factoring (i) the transition type one-step behaviour, represented by a functor F, from (ii) the branching type, represented by a monad T. Using this factorisation, one first aggregates a T-structured collection of a-successors and then updates

*This work was funded by the Advanced Research + Invention Agency (ARIA) through project code MSAI-SE01-P011.

Submitted to:
ACT 2026

this collection through a lifted transition map compatible with the original transition type specified by the functor F. The traditional subset construction is a special case of this framework where the transition type is that of a Moore machine of a specific kind and the branching is given by the finite powerset monad $\mathrm{T}(-) = \mathcal{P}_{\mathrm{fin}}(-)$. For this monad, determinisation groups states (by union) that can be reached by consuming the same input. Its extension to (Rabin) probabilistic automata follows a similar pattern, but with the distribution monad with finite support $\mathrm{T}(-) = \mathrm{D}(-)$. In this case, determinisation averages transitions by a given distribution of initial states, i.e. it performs a pushforward of probabilities along the transitions of the given coalgebra. One consequence of this construction is that, when final coalgebras exist, determinisation induces a behavioural semantics on determinised coalgebras. For the examples above, this semantics is indexed by finite input words. In particular, when $\mathrm{T} = \mathrm{D}$, the determinised coalgebra lives on $\mathrm{D}(X)$, and for each $n \geq 0$ assigns to every input word $w \in I^n$ a distribution in $\mathrm{D}(O)$, obtained by propagating an initial distribution on states through the transitions induced by w. Thus the semantics records the distribution of the current output after a finite input history has been processed.

This semantics is natural when only the current output matters. However, there are settings in which one also needs, for each possible realised output, the corresponding update of the state distribution, so that prediction or control can be implemented consistently along an observed trajectory. In the probabilistic case, this leads again to the state space $\mathrm{D}(X)$, now interpreted as a space of *beliefs* updated by Bayesian filtering [Kal60, Jaz70]. In control and reinforcement learning this yields the *belief MDP* where posterior distributions over hidden states form a fully observable state space for prediction and decision making, and their transition is given by the Bayes filter [Åst65, Str65, KLC98, SSSM22]. Concretely, given a prior $p \in \mathrm{D}(X)$ and an input $i \in I$, one forms a joint distribution on $O \times X$ of next output and next state, then takes its marginal on O, and for each possible output o obtains a posterior distribution on next states conditioned on observing o. The same construction appears in computational mechanics as the *mixed-state presentation* [EMC09, CER16], see [RBB25, Section 5.1]. There, finite observation histories induce distributions over the (hidden) states of a stochastic generator, called mixed states. This treatment highlights an important aspect of this process, *unifilarity*: once the current mixed state and the realised observation are fixed, the next mixed state is uniquely determined. Recent work formulates this categorically as *unifilarisation*, with a functor from categories of stochastic machines to categories of unifilar machines [Vir23].

In this paper, we show that unifilarisation is itself an instance of coalgebraic determinisation, but not of the standard Moore-style form. For structured Moore machines over the distribution monad D, determinisation aggregates successor behaviour by averaging and yields final semantics of the form $I^n \to \mathrm{D}(O)$, recording only the distribution of the current output after a finite input history has been processed. By contrast, we work with Mealy machines and monads with support, that is, monads T equipped with isomorphisms of the form $\mathrm{T}(O \times X) \cong \coprod_{p \in \mathrm{T}(O)} \mathrm{T}(X)^{\mathrm{supp}(p)}$. From these we define supported Mealy machines of the functor $\mathrm{F}(X) = (\coprod_{p \in \mathrm{T}(O)} X^{\mathrm{supp}(p)})^I$. Supported Mealy machines then fit the structured-coalgebra pattern as coalgebras $f : X \to \mathrm{F}(\mathrm{T}X)$, and we construct a compatible T-algebra $\xi : \mathrm{T}(\mathrm{F}(\mathrm{T}X)) \to \mathrm{F}(\mathrm{T}X)$. The general determinisation procedure therefore sends a machine with state space X to one with state space $\mathrm{T}(X)$. For $\mathrm{T} = \mathrm{D}$, the resulting determinised coalgebra is exactly the unifilarisation of a stochastic Mealy machine: given a prior and an input, it forms the prior-weighted joint law of next output and next state, then rewrites it as an output marginal together with posterior next-state distributions. These posterior distributions are precisely the same of belief MDPs and mixed-state presentations. This also induces a different final coalgebra semantics from the Moore case: instead of assigning only a distribution on current outputs to each finite input word, it yields causal stochastic behaviours, represented by families $b_n : I^n \to \mathrm{D}(O^n)$ compatible with Bayesian filtering updates.

2 Structured coalgebras

This section provides an overview of the generalised (coalgebraic) determinisation construction approach introduced by [SBBR10, SBBR13], focusing on a few core examples relevant for a comparison of final coalgebra semantics introduced for different kinds of determinations that can be shown to be special cases of this general construction. Following [SBBR10, SBBR13], we introduce structured coalgebras, or FT-coalgebras, coalgebras parametrised by a functor F and a monad T, and carrying a T-algebra structure.

Definition 1 (Structured coalgebras, or FT-coalgebras [SBBR10, SBBR13])**.** Let $\mathbf{C}$ be a category, $\mathrm{F} : \mathbf{C} \to \mathbf{C}$ an endofunctor and (T, η, μ) a monad on $\mathbf{C}$. An FT-coalgebra is a coalgebra $(S, f : S \to \mathrm{FT}(S))$ of the endofunctor FT, together with a morphism $\xi : \mathrm{TFT}(S) \to \mathrm{FT}(S)$ such that $(\mathrm{FT}(S), \xi)$ is a T-algebra over the monad (T, η, μ).

Explicitly, this means a structured coalgebra consists of:

- an object S in $\mathbf{C}$,
- a morphism $f : S \to \mathrm{FT}(S)$, to be seen as a coalgebra of FT,
- a morphism $\xi : \mathrm{TFT}(S) \to \mathrm{FT}(S)$,

such that the following diagrams, representing unit and action properties, commute:

$$\begin{array}{ccc} \mathrm{FT}(S) & \xrightarrow{\eta_{\mathrm{FT}(S)}} & \mathrm{TFT}(S) \\ & \searrow^{\mathsf{id}_{\mathrm{FT}(S)}} & \downarrow{\scriptstyle \xi} \\ & & \mathrm{FT}(S) \end{array} \qquad \begin{array}{ccc} \mathrm{TTFT}(S) & \xrightarrow{\mathrm{T}\xi} & \mathrm{TFT}(S) \\ \downarrow{\scriptstyle \mu_{\mathrm{FT}(S)}} & & \downarrow{\scriptstyle \xi} \\ \mathrm{TFT}(S) & \xrightarrow[\xi]{} & \mathrm{FT}(S) \end{array} \tag{1}$$

The idea is that the monad T models some kind of nondeterminism, and a coalgebra of FT models a process with a nondeterministic component. Typical choices for T are $\mathrm{T} = \mathcal{P}_{\mathrm{fin}}$ for finite possibilities, $\mathrm{T} = \mathrm{D}$ for discrete probabilities, or $\mathrm{T} = (-) + 1$ for partiality. An FT-coalgebra can seen as a system where the next-step behaviour is "F-shaped", and the successor states come with branching or nondeterminism of type T. The generalised determinisation procedure of [SBBR10, SBBR13] takes a structured coalgebra (S, f, ξ) and produces a map $f^\sharp : \mathrm{T}(S) \to \mathrm{FT}(S)$, to be seen as a coalgebra of F with carrier $\mathrm{T}(S)$. This can be seen as a deterministic system because it is a coalgebra of F only, without the nondeterminism.

Construction 2 (Determinisation of FT-coalgebras [SBBR10, SBBR13])**.** A structured coalgebra (S, f, ξ) can be extended uniquely to a homomorphism of T-algebras $f^\sharp : (\mathrm{T}S, \mu_S) \to (\mathrm{F}(\mathrm{T}S), \xi)$. This unique extension is given by

$$\mathrm{T}(S) \xrightarrow{f^\sharp} \mathrm{FT}(S) = \mathrm{T}(S) \xrightarrow{\mathrm{T}f} \mathrm{TFT}(S) \xrightarrow{\xi} \mathrm{FT}(S). \tag{2}$$

The resulting map $f^\sharp$ is a coalgebra of F with carrier $\mathrm{T}(S)$ and is called the *determinisation* of the coalgebra f, or the *determinised* coalgebra.

Intuitively, ξ tells us how to combine a T-collection of one-step behaviours into a single one-step behaviour. Using this, the original coalgebra extends uniquely to $f^\sharp := \xi \circ \mathrm{T}f$, which give the transitions of the determinised coalgebra. In [SBBR10, SBBR13] it is shown that various constructions, including the classical power set construction and the totalisation of partial automata, are instances of this. We will later show that unifilarisation, or the construction of a "belief machine" is also an instance, for an appropriate choice of F, T and ξ.

2.1 Example - Structured Moore machines

Various examples of this parametrisation are given in [SBBR10, SBBR13], including partial Mealy automata, nondeterministic automata, pushdown automata, etc. Here we are especially interested in one of these examples, structured Moore machines, which are structured coalgebras for the functor $\mathrm{F}(-) = \mathrm{T}(O) \times (-)^I$ and an arbitrary monad T. We are interested in these machines because of the final coalgebra semantics they come equipped with for specific monads T, which we will use to compare to the semantics induced by a different choice of F, T and ξ that capture Bayesian updates, see Section 4.3.

To define structured Moore machines we need the following proposition, which is part of the standard theory of monads on **Set**. It relies on the fact that every monad on a Cartesian closed category has a canonical strength.

Proposition 3. *Given a monad (T, η, μ) on a Cartesian closed category, an algebra (X, h) for* T *and an arbitrary set I, the set X^I also carries an algebra structure $h^{(I)} : \mathrm{T}(X^I) \to X^I$ given by currying the map*

$$\mathrm{T}(X^I) \times I \xrightarrow{\mathrm{str}_{X^I, I}} \mathrm{T}(X^I \times I) \xrightarrow{T(\mathrm{eval}_{X,I})} \mathrm{T}(X) \xrightarrow{h} X. \tag{3}$$

Proof. Omitted. (This is part of the standard theory of monads.) □

Definition 4 (Structured Moore machines [SBBR13])**.** Given a monad (T, η, μ) and sets I and O, a T-structured Moore machine is an FT-coalgebra $(S, f_{\mathsf{FTMoore}}, \xi)$, where $\mathrm{F} = \mathrm{T}(O) \times (-)^I$, and $\xi : \mathrm{T}(\mathrm{T}(O) \times \mathrm{T}(S)^I) \to \mathrm{T}(O) \times \mathrm{T}(S)^I$ is given by

$$\xi := \mathrm{T}(\mathrm{T}(O) \times \mathrm{T}(S)^I) \xrightarrow{\langle \mathrm{T}\pi_{\mathrm{T}(O)}, \mathrm{T}\pi_{\mathrm{T}(S)^I} \rangle} \mathrm{T}(\mathrm{T}(O)) \times \mathrm{T}(\mathrm{T}(S)^I) \xrightarrow{\mu_O \times \mu_S^{(I)}} \mathrm{T}(O) \times \mathrm{T}(S)^I \tag{4}$$

where π_i are projections for the i'th component of the Cartesian product and $\mu_S^{(I)}$ is Proposition 3 applied to the free algebra $\mathrm{T}(S)$.

We will sometimes refer to the map $f_{\mathsf{FTMoore}} : S \to \mathrm{T}(O) \times \mathrm{T}(S)$ by its components,

$$f_{\mathsf{FTMoore}} = \langle \mathsf{out}_{\mathsf{FTMoore}} : S \to \mathrm{T}(O),\ \mathsf{tr}_{\mathsf{FTMoore}} : S \to \mathrm{T}(S)^I \rangle. \tag{5}$$

For particular choices of T, we can use this definition to obtain nondeterministic Moore machines (for the finite nonempty powerset monad, $\mathcal{P}_{\mathrm{fin}}$) and stochastic Moore machines (for the distribution monad, D). Taking the distribution monad as an example, we can regard a structured Moore machine as a machine that, on each time step, produces an output in O stochastically, as well as taking in an input from I and then stochastically updating its state. Since structured Moore machines are structured coalgebras, they can be determinised using Construction 2, yielding the following.

Proposition 5 (Determinised structured Moore machines [SBBR13])**.** *Let $(S, f_{\mathsf{FTMoore}} : S \to \mathrm{T}(O) \times \mathrm{T}(S)^I)$ be a structured Moore machine, understood as an* FT*-coalgebra. By determinising this structured Moore machine, we obtain a coalgebra $(\mathrm{T}(S), f^\sharp_{\mathsf{FTMoore}} : \mathrm{T}(S) \to \mathrm{T}(O) \times \mathrm{T}(S)^I)$, with $f^\sharp_{\mathsf{FTMoore}}$ given by*

$$f^\sharp_{\mathsf{FTMoore}} = \langle \mathsf{out}^\sharp_{\mathsf{FTMoore}} : \mathrm{T}(S) \to \mathrm{T}(O),\ \mathsf{tr}^\sharp_{\mathsf{FTMoore}} : \mathrm{T}(S) \to \mathrm{T}(S)^I \rangle, \tag{6}$$

where

$$\mathsf{out}^\sharp_{\mathsf{FTMoore}} = \mathrm{T}(S) \xrightarrow{\mathrm{T}(f_{\mathsf{FTMoore}})} \mathrm{T}(\mathrm{T}(O) \times \mathrm{T}(S)^I) \xrightarrow{\mathrm{T}(\pi_{\mathrm{T}(O)})} \mathrm{T}(\mathrm{T}(O)) \xrightarrow{\mu_O} \mathrm{T}(O) \tag{7}$$

and

$$\mathsf{tr}^\sharp_{\mathsf{FTMoore}} = \mathrm{T}(S) \xrightarrow{\mathrm{T}(f_{\mathsf{FTMoore}})} \mathrm{T}(\mathrm{T}(O) \times \mathrm{T}(S)^I) \xrightarrow{\mathrm{T}(\pi_{\mathrm{T}(S)^I})} \mathrm{T}(\mathrm{T}(S)^I) \xrightarrow{\mu_S^{(I)}} \mathrm{T}(S)^I. \tag{8}$$

Proof. Straightforward. □

In the case of the distribution monad, this produces a machine whose state space is $\mathrm{D}(S)$. The output map $\mathsf{out}^\sharp_{\mathsf{FTMoore}}$ takes as input a measure over S and pushes it forward along the Markov kernel $S \xrightarrow{f_{\mathsf{FTMoore}}} \mathrm{D}(\mathrm{D}(O)) \times \mathrm{D}(S)^I) \xrightarrow{\mathrm{T}(\pi_{\mathrm{D}(O)})} \mathrm{D}(\mathrm{D}(O)) \xrightarrow{\mu_O} \mathrm{D}(O)$. The transition map $\mathsf{tr}^\sharp_{\mathsf{FTMoore}}$ does something similar, pushing forward a measure over S along the map $\mathrm{T}(\pi_{\mathrm{T}(S)^I}) \circ f^\sharp_{\mathsf{FTMoore}}$ to get a measure over functions $I \to \mathrm{T}(S)$ (that is, an element of $\mathrm{T}(\mathrm{T}(S)^I)$). The $\mu^{(I)}_S$ step can be seen as taking an input in I and passing it to each of these functions to get an element of $\mathrm{T}(\mathrm{T}(S))$, and then averaging to get an element of $\mathrm{T}(S)$. But this doesn't involve conditioning on an observation, since no sample from the output distribution is considered.

From the perspective of someone interested in reinforcement learning and POMDPs however, it might seem surprising that these dynamics involve pushing forward a measure along a Markov kernel but that they don't involve a Bayesian conditioning step. The reason for this becomes clearer when considering the final coalgebra semantics. A final coalgebra is said to capture the behaviour of a coalgebra [Rut00, Jac17]. More precisely, the elements of a final coalgebra (when it exists) are the possible observable behaviours of all objects of a given category of coalgebras.

Following standard arguments from automata theory, [SBBR10, SBBR13] argue that for FT-coalgebras, we should consider behaviours to be the final coalgebra of F rather than FT, so that the behaviours of a FT-coalgebra are the F-behaviours of its determinisation. This is because the final coalgebra of FT is too fine-grained: it can consider two machines to have different behaviours even if they are indistinguishable to an external observer, because they differ in the behaviour of their internal state. Intuitively, determinisation 'absorbs' these internal differences.

We now consider the final coalgebra semantics in this sense for determinised stochastic Moore machines, which are structured Moore machines with $\mathrm{T} = \mathrm{D}$. The functor F in this case is $\mathrm{D}(O) \times (-)^I$.

Proposition 6 (F-behaviours for stochastic Moore machines [SBBR10, SBBR13])**.** *The endofunctor* $\mathrm{D}(O) \times (-)^I$ *has final coalgebra:*

$$\left(\mathrm{D}(O)^{I^\star}, \omega^\sharp_{\mathsf{FTMoore}} : \mathrm{D}(O)^{I^\star} \to \mathrm{D}(O) \times \left(\mathrm{D}(O)^{I^\star}\right)^I\right). \tag{9}$$

Elements of the carrier of this final coalgebra are maps $g : I^\star \to \mathrm{D}(O)$ assigning to each finite input word $w \in I^\star$ a distribution $g(w) \in \mathrm{D}(O)$ on observations. Equivalently, for each $o \in O$, the quantity $g(w)(o) \in \mathrm{D}(O)$ is the probability of observing o after reading w. This generalises the usual semantics of (Rabin) probabilistic automata [Rab63]: when $O = 2$, each distribution $g(w) \in \mathrm{D}(2)$ is determined by a single scalar in $[0,1]$, namely the probability assigned to the designated accepting output, so one recovers the standard acceptance probability of the word w. We note that the output here is stochastic, but it only depends on the sequence of previous inputs. This makes sense in terms of Rabin probabilistic automata, where the idea is to give a word as input to the machine, and obtain as output the probability of accepting or rejecting the word. Since we only ever consider one output in this situation there is no need to invoke conditioning via Bayes, since there are no observations to condition on, besides the last output.

However, in different situations involving agents solving, e.g. POMDPs in reinforcement learning and control theory, one is interested not only in the final output given a sequence of inputs, but in the whole sequence of intermediate outputs that the machine emits as well. This is because a controller can observe these intermediate outputs from the POMDP and might choose to give different inputs depending on what they are. For this reason, we desire different semantics in a POMDP setting. It turns out that this

can be achieved in the determinisation framework of [SBBR13], but we must choose a different class of automata. This will be the topic of the rest of the paper.

3 Monads with support

For the purpose of this paper, and particularly for the instantiation of unifilarisation in coalgebraic terms following [Vir23], it will be more convenient from now on to work with Mealy machines rather than Moore machines. We will however need Mealy machines of a particular kind, so before defining them formally, we introduce the notion of monad with support and a few examples of it.

Monads with support are monads T that come with some extra structure, which we call *support*, because in the case of the distribution monad it corresponds to the support of a distribution. This will be important because for monads with support we will obtain the finitely supported version of the definition of unifilarisation given in [Vir23], which will then also be derived from the generalised determinisation construction of [SBBR13].

As in the case of the generalised determinisation of [SBBR10, SBBR13], we restrict our attention here to monads on **Set**. This means that we do not attempt to extend Definition 7 to measure-theoretic treatments. The notion of monad with support will probably not directly transfer to such contexts, since the notion of support is quite subtle in category-theoretic probability [FGL+26].

Definition 7 (Monad with support)**.** A *monad with support* consists of a monad T on **Set** together with, for each X and each $p \in \mathrm{T}(X)$, a subset $\mathrm{supp}(p) \subseteq X$ called the *support* of p, and for each X, Y an isomorphism

$$\Phi_{X,Y} : \mathrm{T}(X \times Y) \xrightarrow{\cong} \coprod_{p \in \mathrm{T}(X)} \mathrm{T}(Y)^{\mathrm{supp}(p)}, \tag{10}$$

natural in both X and Y.

By naturality in both X and Y we mean that the assignment $(X,Y) \mapsto \coprod_{p\in\mathrm{T}(X)} \mathrm{T}(Y)^{\mathrm{supp}(p)}$ extends to a bifunctor $G : \mathbf{Set} \times \mathbf{Set} \to \mathbf{Set}$ given on objects by

$$G(X,Y) := \coprod_{p \in \mathrm{T}(X)} \mathrm{T}(Y)^{\mathrm{supp}(p)}, \tag{11}$$

and, on morphisms $f : X \to X'$ and $g : Y \to Y'$ by,

$$G(f,g) := \Phi_{X',Y'} \circ \mathrm{T}(f \times g) \circ \Phi^{-1}_{X,Y}, \tag{12}$$

so that the following diagram commutes

$$\begin{array}{ccc} \mathrm{T}(X \times Y) & \xrightarrow{\Phi_{X,Y}} & G(X,Y) \\ \Big\downarrow{\scriptstyle \mathrm{T}(f\times g)} & & \Big\downarrow{\scriptstyle G(f,g)} \\ \mathrm{T}(X' \times Y') & \xrightarrow[\Phi_{X',Y'}]{} & G(X',Y') \end{array} \tag{13}$$

By construction, G satisfies $G(\mathsf{id}_X, \mathsf{id}_Y) = \mathsf{id}_{G(X,Y)}$ and $G(f',g') \circ G(f,g) = G(f' \circ f, g' \circ g)$. Thus the family $(\Phi_{X,Y})_{X,Y}$ is a natural isomorphism $\Phi : \mathrm{T}(- \times -) \xRightarrow{\cong} G$.

The definition of supp depends on the specific monad. We give some examples below, namely the distribution monad and the finite nonempty powerset monad. Examples that don't fit the definition (at least not in the most obvious nontrivial way) include the powerset monad (finite or not), the partiality monad and the subdistribution monad.

3.1 Examples of monads with support

Example 8. The distribution monad D is a monad with support.

To see this, we give some definitions from elementary probability theory, in the language of the distribution monad. In the following, given $p \in \mathrm{D}(X)$ and $x \in X$, we write $p(x)$ for the probability assigned to x according to the distribution p, and similarly we write $p'(x,y)$ for the probabilities assigned by a distribution $p' \in \mathrm{D}(X \times Y)$.

Definition 9 (Support of a probability distribution)**.** Let D be the distribution monad. Given a set X, let the map $\mathrm{supp}_X : \mathrm{D}(X) \to \mathbf{Set}$ be given by $\mathrm{supp}_X(p) = \{x \in X \mid p(x) > 0\}$. Since there is little danger of ambiguity we will just write this as $\mathrm{supp}(p)$.

Definition 10 (Marginal probability)**.** Given an element p of $\mathrm{D}(X \times Y)$ (seen as a joint distribution between X and Y), define the *marginal over* X as $p_X \in \mathrm{D}(X)$, where $p_X(x) = \sum_y p(x,y)$.

This is well defined because since p is finitely supported then there are only finitely many $x \in X$ where $p_X(x)$ is nonzero, so p_X is finitely supported as well. We now define conditional probability distributions, restricting their domain in order to avoid dividing by zero.

Definition 11 (Conditional probability)**.** Given $p \in \mathrm{D}(X \times Y)$ and $x \in \mathrm{supp}\, p_X$, define the *conditional distribution over* Y *given* x as $p_{Y|x} \in \mathrm{D}(Y)$, where $p_{Y|x}(y) = p(x,y)/p_X(x)$.

We can again note that this is well defined, because since p is finitely supported, there are only finitely many y for which $p_{Y|x}(y) > 0$. We can then express a basic fact about probability:

Lemma 12. *Given sets X and Y, there is an isomorphism, natural in both X and Y*

$$\Phi_{X,Y} : \mathrm{D}(X \times Y) \xrightarrow{\cong} \coprod_{p \in \mathrm{D}(X)} \mathrm{D}(Y)^{\mathrm{supp}(p)} \tag{14}$$

given in the $\mathrm{D}(X \times Y) \to \coprod_{p \in \mathrm{D}(X)} \mathrm{D}(Y)^{\mathrm{supp}(p)}$ direction by "disintegrating" [CJ19] the distribution into a marginal and a conditional:

$$p \mapsto (p_X \,,\, \lambda x \,.\, p_{Y|x}). \tag{15}$$

The inverse in the $\coprod_{p \in \mathrm{D}(X)} \mathrm{D}(Y)^{\mathrm{supp}(p)} \to \mathrm{D}(X \times Y)$ direction maps $(q_X \,,\, \lambda x \,.\, q_{Y|x})$ to $q \in \mathrm{D}(X \times Y)$, where $q(x,y) \coloneqq q_X(x) q_{Y|x}(y)$.

Proof. Omitted. (This is a well known fact about elementary probability theory.) □

In the following, $\mathcal{P}^+_{\mathrm{fin}}$ refers to the nonempty powerset monad, which is the same as the usual finite powerset monad $\mathcal{P}_{\mathrm{fin}}$, except that we exclude the empty set, so that $\mathcal{P}^+_{\mathrm{fin}}(X) = \{U \subseteq X \mid U \text{ is finite and } U \neq \varnothing\}$.

Example 13. The finite nonempty powerset monad $\mathcal{P}^+_{\mathrm{fin}}$ is a monad with support, where $\mathrm{supp}(U) = U$.

That is, since $U \in \mathcal{P}^+_{\text{fin}}$ is already a set, $U \subseteq X$, we can take the support to be U itself. The isomorphism

$$\Phi_{X,Y} : \mathcal{P}^+_{\text{fin}}(X \times Y) \xrightarrow{\cong} \coprod_{A \in \mathcal{P}^+_{\text{fin}}(X)} \mathcal{P}^+_{\text{fin}}(Y)^{\text{supp}(A)}. \tag{16}$$

is then given in the forward direction by

$$U \mapsto \left(\{x \in X \mid (x,y) \in U\}, \lambda x.\, \{y \in Y \mid (x,y) \in U\}\right) \tag{17}$$

and in the reverse direction by

$$(U_X, f) \mapsto \{(x,y) \in X \times Y \mid x \in U_X,\ y \in f(x)\}. \tag{18}$$

Remark 14. Another example of a monad with support is the nonempty powerset monad $\mathcal{P}^+$, since nothing above requires the finiteness condition. However, like the powerset monad, the nonempty powerset monad lacks a final coalgebra.

4 Unifilarisation as determinisation of Mealy machines

4.1 Mealy machines

As hinted at before, we now formally switch from Moore machines to Mealy machines because unifilarisation is naturally formulated in terms of one-step joint behaviour of outputs and next states. This is the key structural difference from the Moore case: in a Mealy coalgebra $S \to \mathrm{T}(O \times S)^I$, the output and next state are packaged together, so after averaging one can rewrite the resulting joint distribution as an output marginal together with posterior next-state distributions. That rewriting step is what later produces Bayesian updates. We thus initially introduce Mealy machines as coalgebras following [Rut00, Jac17]. To capture existing definitions that include deterministic, nondeterministic, and stochastic Mealy machines [Vir23, BDLR25], we parametrise our definition with a monad T.

Definition 15 (Mealy machines)**.** Mealy machines are coalgebras of the functor

$$\mathrm{F}_{\mathsf{TMealy}}(-) = \mathrm{T}(O \times -)^I. \tag{19}$$

Using monads with support Definition 7, we then introduce the following notion.

Definition 16 (Supported Mealy machines)**.** Given a monad with support T, supported Mealy machines are coalgebras of the functor

$$\mathrm{F}_{\mathsf{TSuppMealy}}(-) = \left(\coprod_{p \in \mathrm{T}(O)} \mathrm{T}(-)^{\text{supp}(p)}\right)^I. \tag{20}$$

A supported Mealy machine can be seen as a gadget that starts in some state in X, receives an input in I, and then produces both an output in O and its own next state in X under the branching specified by the monad T. In general the output and the next state can be correlated, given the previous state and the input. Calling these "Mealy machines" is justified by Eq. (10), which implies that $\mathrm{F}_{\mathsf{TSuppMealy}} \cong \mathrm{F}_{\mathsf{TMealy}} = T(O \times -)^I$.

Remark 17. Since we will only consider supported Mealy machines from now on, we shall refer to supported Mealy machines simply as Mealy machines.

An important example of Mealy machines is that of stochastic Mealy machines.

Definition 18 (Stochastic Mealy machines)**.** Stochastic Mealy machines are coalgebras of the functor $\mathrm{F}_{\mathsf{DSuppMealy}}$, i.e. Mealy machines for $\mathrm{T} = \mathrm{D}$.

Next we show how unifilarisation can be expressed for Mealy machines and in particular for stochastic ones, which capture the definition by [Vir23].

4.2 Unifilarisation of Mealy machines

Here we adapt the unifilarisation definition given in [Vir23] to our framework, in which Mealy machines are seen as coalgebras. To do that, we first introduce *unifilar* Mealy machines. These are a variation on the concept of a Mealy machine, which is somewhat less intuitive but is closely related to a central concept in computational mechanics, that of a unifilar hidden Markov process [SC01, BC15].

Definition 19 (Unifilar Mealy machines)**.** Unifilar Mealy machines are coalgebras of the (polynomial) functor

$$\mathrm{F}_{\mathsf{UnifTMealy}}(-) = \left(\coprod_{p \in \mathrm{T}(O)} (-)^{\mathrm{supp}(p)} \right)^{I}. \tag{21}$$

A unifilar Mealy machine can be seen as a gadget that starts in some state in X, takes an input in I, generates an output in O with branching specified by T, and then *deterministically* updates its state, as a function of both the given input and its own generated output, which can be stochastic, nondeterministic, etc. Its next state is only defined on those outputs that it produces with positive probability, given its current state and input.

We will show shortly that, for $\mathrm{T} = \mathrm{D}$, Mealy machines can be given a T-algebra structure such that the determinisation of a Mealy machine is a unifilar Mealy machine. Before we do that, however, we first spell out what the definition of unifilarisation from [Vir23] amounts to when specialised to the distribution monad, in our current notation. We will later show that determinisation produces the same result.

Definition 20 (Unifilarisation of stochastic Mealy machines [Vir23])**.** Let $(S, f_{\mathsf{DSuppMealy}})$ be a stochastic Mealy machine. Its unifilarisation is the unifilar Mealy machine $(\mathrm{D}(S), f_{\mathsf{UnifDMealy}})$ where transitions

$$f_{\mathsf{UnifDMealy}} : \mathrm{D}(S) \to \left(\coprod_{p \in \mathrm{D}(O)} \mathrm{D}(S)^{\mathrm{supp}(p)} \right)^{I} \tag{22}$$

are defined as follows. For $q \in \mathrm{D}(S)$ and $i \in I$, first define the joint distribution

$$\rho_{q,i}(o, s') \coloneqq \sum_{s \in S} q(s)\, \Phi^{-1}_{O,S}\left(f_{\mathsf{DSuppMealy}}(s)(i) \right)(o, s'). \tag{23}$$

This takes a little unpacking. The idea is that for each outcome s, $f_{\mathsf{DSuppMealy}}(s)(i)$ is an element of $\left(\coprod_{p \in \mathrm{D}(O)} \mathrm{T}(S)^{\mathrm{supp}(p)} \right)$, and so we can turn it into a joint distribution over $O \times S$ via the isomorphism $\Phi^{-1}_{O,S}$. We then take the average of all of these. Conceptually, all we are doing here is calculating the expected distribution over outputs s', o of the Mealy machine, given an input i and a distribution q over initial states.

Next we define the output marginal over O and, for each $o \in \mathrm{supp}(p_{q,i})$, the posterior next-state distribution:

$$p_{q,i}(o) \coloneqq \sum_{s' \in S} \rho_{q,i}(o, s') \qquad k_{q,i}(o)(s') \coloneqq \frac{\rho_{q,i}(o, s')}{p_{q,i}(o)}. \tag{24}$$

We then set $f_{\mathsf{UnifDMealy}}(q)(i) \coloneqq (p_{q,i}, k_{q,i})$.

Unifilarisation produces an epistemic model, a unifilar Mealy machine, from a given stochastic Mealy machine that could (but doesn't have to) represent an environment for an agent. It is called "epistemic" because the state space of the unifilar Mealy machine is given by *beliefs* about X, $\mathrm{D}(X)$, that could be said to belong to an idealised Bayesian reasoner, or to an agent. The idea is that the reasoner's prior at any given time is an element of the state space $\mathrm{D}(X)$, which is updated following Bayesian filtering. This process underlies the construction of belief MDPs in reinforcement learning [KLC98] and their previous formulations in terms of state information and sufficient statistics for control in control theory [Åst65, Str65], which correspond to unifilar Mealy machines with marginalised observations [Vir23]. It also generalises the construction of mixed state presentations in computational mechanics [EMC09, CER16]. For $\mathrm{T} = \mathcal{P}^+_{\mathrm{fin}}$, we obtain the same construction proposed by [Vir23] since the Kleisli category of the (finite) nonempty powerset monad is also a strongly representable Markov category.

Next we can note that $\mathrm{F}_{\mathsf{TSuppMealy}} = \mathrm{F}_{\mathsf{UnifTMealy}}\mathrm{T}$, so that (supported) Mealy machines are coalgebras of $\mathrm{F}_{\mathsf{UnifTMealy}}\mathrm{FT}$. After showing that coalgebras of this type carry a T-algebra structure, we will then prove that the determinisation construction applied to these coalgebras will give unifilar machines implementing Bayesian updates.

4.3 Determinisation of Mealy machines

Mealy machines can be equipped with a T-algebra structure. To do so, we first give $\coprod_{t\in\mathrm{T}(O)} \mathrm{T}(S)^{\mathrm{supp}(t)}$ an algebra structure.

Proposition 21. *The map* $\nu : \mathrm{T}\left(\coprod_{t\in\mathrm{T}(O)} \mathrm{T}(S)^{\mathrm{supp}(t)}\right) \to \coprod_{t\in\mathrm{T}(O)} \mathrm{T}(S)^{\mathrm{supp}(t)}$, *given by*

$$\nu := \mathrm{T}\left(\coprod_{t\in\mathrm{T}(O)} \mathrm{T}(S)^{\mathrm{supp}(t)}\right) \xrightarrow{\mathrm{T}(\Phi^{-1}_{O,S})} \mathrm{T}\left(\mathrm{T}(O\times S)\right) \xrightarrow{\mu_{O\times S}} \mathrm{T}(O\times S) \xrightarrow{\Phi_{O,S}} \coprod_{t\in\mathrm{T}(O)} \mathrm{T}(S)^{\mathrm{supp}(t)}, \tag{25}$$

induces an algebra structure on $\coprod_{t\in\mathrm{T}(O)} \mathrm{T}(S)^{\mathrm{supp}(t)}$.

Proof. This is the transport of the free algebra structure on $\mathrm{T}(O\times S)$ along the isomorphism Φ. □

Then $\mathrm{F}_{\mathsf{TSuppMealy}}(S) = \left(\coprod_{t\in\mathrm{T}(O)} \mathrm{T}(S)^{\mathrm{supp}(t)}\right)^I$ has an algebra structure given by

$$\xi := \mathrm{T}\left(\left(\coprod_{t\in\mathrm{T}(O)} \mathrm{T}(S)^{\mathrm{supp}(t)}\right)^I\right) \xrightarrow{\nu^{(I)}} \left(\coprod_{t\in\mathrm{T}(O)} \mathrm{T}(S)^{\mathrm{supp}(t)}\right)^I, \tag{26}$$

which is an algebra structure by Proposition 3.

Since Mealy machines with this T-algebra structure are FT-algebras, they can be determinised using [SBBR13]'s construction, which yields a determinised machine

$$\left(\mathrm{T}(S), f^\sharp_{\mathsf{TSuppMealy}} = \xi \circ \mathrm{T} f_{\mathsf{TSuppMealy}}\right) \tag{27}$$

with ξ as in Eq. (26).

As our main result, we now show that in the case $\mathrm{T} = \mathrm{D}$, these determinised machines are exactly the ones given by the unifilarisation of [Vir23] in the form of Definition 20.

Theorem 22 (Unifilarisation is determinisation)**.** *Let* $(S, f_{\mathsf{DSuppMealy}} : S \to \left(\coprod_{p\in \mathrm{D}(O)} \mathrm{D}(S)^{\mathrm{supp}(p)}\right)^I$ *be a stochastic Mealy machine, equipped with the algebra map in Eq.* (26)*, with* $\mathrm{T} = \mathrm{D}$*. Its determinisation is exactly its unifilarisation according to Definition 20.*

Sketch proof. The determinisation of $(S, f_{\mathsf{DSuppMealy}})$ is given by $(\mathrm{D}(S), f^\sharp_{\mathsf{DSuppMealy}})$, where, using Eqs. (26) and (27),

$$f^\sharp_{\mathsf{DSuppMealy}} = \mathrm{D}(S) \xrightarrow{\mathrm{D}(f_{\mathsf{DSuppMealy}})} \mathrm{D}\left(\left(\coprod_{p\in \mathrm{D}(O)} \mathrm{D}(S)^{\mathrm{supp}(p)}\right)^I\right) \xrightarrow{\nu^{(I)}} \left(\coprod_{p\in \mathrm{D}(O)} \mathrm{D}(S)^{\mathrm{supp}(p)}\right)^I. \tag{28}$$

We wish to show that, for $q \in \mathrm{D}(S)$ and $i \in I$, we have $f^\sharp_{\mathsf{DMealy}}(q)(i) = (p_{q,i}, k_{q,i})$, with $p_{q,i}$ and $k_{q,i}$ as defined in Eq. (24).

To proceed, we uncurry $\nu^{(I)}$ and expand the definition of ν noting that $f^\sharp_{\mathsf{DSuppMealy}}$ is the curried version of

$$\begin{aligned}\mathrm{D}(S)\times I &\xrightarrow{\mathrm{D}(f_{\mathsf{DSuppMealy}})\times \mathsf{id}_I} \mathrm{D}\left(\left(\coprod_{t\in \mathrm{D}(O)} \mathrm{D}(S)^{\mathrm{supp}(t)}\right)^I\right)\times I \xrightarrow{\mathrm{str}_{(\cdot)^I, I}} \mathrm{D}\left(\left(\coprod_{t\in \mathrm{D}(O)} \mathrm{D}(S)^{\mathrm{supp}(t)}\right)^I \times I\right)\\ &\xrightarrow{\mathrm{D}(\mathrm{eval}_{(\cdot),I})} \mathrm{D}\left(\coprod_{t\in \mathrm{D}(O)} \mathrm{D}(S)^{\mathrm{supp}(t)}\right) \xrightarrow{T(\Phi^{-1}_{O,S})} \mathrm{T}(\mathrm{T}(O\times S)) \xrightarrow{\mu_{O\times S}} \mathrm{T}(O\times S) \xrightarrow{\Phi_{O,S}} \coprod_{t\in \mathrm{D}(O)} \mathrm{D}(S)^{\mathrm{supp}(t)}.\end{aligned} \tag{29}$$

The first five steps of Eq. (29) are performing the same calculation as Eq. (23), taking as input $q \in DS$ and $i \in I$ and forming the distribution $\rho_{q,i}(o, s')$. The last step forms the marginal $p_{q,i}(o)$ and the conditional $k_{q,i}(o)(s')$, as in Eq. (24), via the isomorphism in Lemma 12. □

As with stochastic Moore machines in Section 2.1, it is helpful to compare this to the final coalgebra semantics. The final coalgebra for unfilar machines is worked out in [Vir23]. We summarise it here, starting with the following definition.

Definition 23 (Causal stochastic behaviour)**.** Let D be the distribution monad, and let $X, Y \in$ **Set**. For each $n \geq 0$, write X^n and Y^n for the n-fold Cartesian products (with $X^0 = Y^0 = 1$), and let

$$\pi^X_n : X^{n+1} \to X^n, \qquad \pi^Y_n : Y^{n+1} \to Y^n \tag{30}$$

be the prefix projections: for $n = 0$, these are the unique maps $\pi^X_0 : X \to 1$ and $\pi^Y_0 : Y \to 1$, and for $n \geq 1$ they are given by $\pi^X_n(x_0, \ldots, x_n) = (x_0, \ldots, x_{n-1})$, $\pi^Y_n(y_0, \ldots, y_n) = (y_0, \ldots, y_{n-1})$. A causal stochastic behaviour from X to Y is a family

$$b_n : X^n \to \mathrm{D}(Y^n) \qquad (n \geq 0) \tag{31}$$

such that

$$\mathrm{D}(\pi^Y_n) \circ b_{n+1} = b_n \circ \pi^X_n \qquad \forall n \geq 0 \tag{32}$$

We write $\mathrm{Caus}_{\mathrm{D}}(X, Y)$ for the set of all such families.

These are called "causal" because each distribution of the first n output symbols depends only on the first n input symbols. Using this fact, we can now define the following.

Proposition 24 (Final coalgebra of determinised stochastic Mealy machines [Vir23])**.** *The endofunctor* $\left(\coprod_{p\in \mathrm{D}(O)}(-)^{\mathrm{supp}(p)}\right)^I$ *has final coalgebra:*

$$\left(\mathrm{Caus}_{\mathrm{D}}(I,O), \omega^{\sharp}_{\mathsf{DMealy}} : \mathrm{Caus}_{\mathrm{D}}(I,O) \to \left(\sum_{p\in \mathrm{D}(O)} (\mathrm{Caus}_{\mathrm{D}}(I,O))^{\mathrm{supp}(p)}\right)^I\right). \tag{33}$$

Elements of $\mathrm{Caus}_{\mathrm{D}}(I,O)$ are causal stochastic behaviours. Such behaviours assign to each finite input word $w \in I^n$ a distribution $b_n(w) \in \mathrm{D}(O^n)$ on output words of length n and these assignments are compatible under marginalisation:

$$\mathrm{D}(\pi^O_n)\,(b_{n+1}(w)) = b_n\left(\pi^I_n(w)\right) \qquad (w \in I^{n+1}). \tag{34}$$

Unlike the Moore case, this semantics records full causal input-output behaviour, rather than only the last output distribution after a finite input word. This gives some intuition for why Bayesian conditioning appears in the determinisation of supported Mealy machines but not for stochastic Moore machines: in order to correctly predict future outputs, the determinised machine needs to take account of what the previous output was, and Bayesian conditioning is the natural way to achieve this.

5 Conclusion

The main result of this paper is that unifilarisation fits the general pattern of coalgebraic determinisation. The point is not that it coincides with the standard Moore-style construction, but that it arises from the same determinisation principle once the transition type is changed appropriately. By passing from Moore machines to Mealy machines over monads with support, we obtain a determinisation procedure on $\mathrm{T}(X)$ that, in the case of the distribution monad, reproduces exactly the Bayesian filtering update of stochastic unifilar machines.

This also clarifies the source of the semantic gap between the two settings. Standard determinisation of stochastic Moore machines keeps track only of the distribution of the current output after a finite input history, in line with the idea of language acceptance from automata theory. Unifilarisation, by contrast, keeps track of output histories together with the posterior updates they induce, and therefore yields a semantics in terms of causal stochastic behaviours, which is more in line with agents solving problems such as POMDPs in reinforcement learning. The contribution of the paper is therefore twofold: it identifies unifilarisation as an instance of coalgebraic determinisation, and it explains precisely why this instance induces a richer behavioural semantics than the standard stochastic Moore case.

References

[Åst65] K.J Åström (1965): *Optimal Control of Markov Processes with Incomplete State Information*. *Journal of Mathematical Analysis and Applications* 10(1), pp. 174–205, doi:10.1016/0022-247X(65)90154-X.

[BC15] Nix Barnett & James P. Crutchfield (2015): *Computational Mechanics of Input–Output Processes: Structured Transformations and the ε-Transducer*. *Journal of Statistical Physics* 161(2), pp. 404–451, doi:10.1007/s10955-015-1327-5.

[BDLR25] Filippo Bonchi, Elena Di Lavore & Mario Román (2025): *Effectful Mealy Machines: Coalgebraic and Causal Traces*. *LIPIcs, Volume 342, CALCO 2025* 342, pp. 1:1–1:7, doi:10.4230/LIPICS.CALCO.2025.1.

[BSS21] Filippo Bonchi, Alexandra Silva & Ana Sokolova (2021): *Distribution Bisimilarity via the Power of Convex Algebras*. *Logical Methods in Computer Science* Volume 17, Issue 3, p. 6158, doi:10.46298/lmcs-17(3:10)2021.

[CER16] James P. Crutchfield, Christopher J. Ellison & Paul M. Riechers (2016): *Exact Complexity: The Spectral Decomposition of Intrinsic Computation*. *Physics Letters A* 380(9), pp. 998–1002, doi:10.1016/j.physleta.2016.01.008.

[CJ19] Kenta Cho & Bart Jacobs (2019): *Disintegration and Bayesian Inversion via String Diagrams*. *Mathematical Structures in Computer Science* 29(7), pp. 938–971, doi:10.1017/S0960129518000488. arXiv:1709.00322.

[EMC09] Christopher J. Ellison, John R. Mahoney & James P. Crutchfield (2009): *Prediction, Retrodiction, and the Amount of Information Stored in the Present*. *Journal of Statistical Physics* 136(6), pp. 1005–1034, doi:10.1007/s10955-009-9808-z.

[FGL+26] Tobias Fritz, Tomáš Gonda, Antonio Lorenzin, Paolo Perrone & Dario Stein (2026): *Absolute Continuity, Supports and Idempotent Splitting in Categorical Probability*, doi:10.48550/arXiv.2308.00651. arXiv:2308.00651.

[Jac17] Bart Jacobs (2017): *Introduction to Coalgebra: Towards Mathematics of States and Observation*, 1 edition. Cambridge University Press, doi:10.1017/CBO9781316823187.

[Jaz70] Andrew H Jazwinski (1970): *Stochastic Processes and Filtering Theory*. 64, Academic Press.

[JSS15] Bart Jacobs, Alexandra Silva & Ana Sokolova (2015): *Trace Semantics via Determinization*. *Journal of Computer and System Sciences* 81(5), pp. 859–879, doi:10.1016/j.jcss.2014.12.005.

[Kal60] Rudolf E Kalman (1960): *A New Approach to Linear Filtering and Prediction Problems*. *Journal of Basic Engineering* 82(1), pp. 35–45.

[KLC98] Leslie P. Kaelbling, Michael L. Littman & Anthony R. Cassandra (1998): *Planning and Acting in Partially Observable Stochastic Domains*. *Artificial Intelligence* 101(1-2), pp. 99–134, doi:10.1016/S0004-3702(98)00023-X.

[Rab63] Michael O. Rabin (1963): *Probabilistic Automata*. *Information and Control* 6(3), pp. 230–245, doi:10.1016/S0019-9958(63)90290-0.

[RBB25] Fernando Rosas, Alexander Boyd & Manuel Baltieri (2025): *AI in a Vat: Fundamental Limits of Efficient World Modelling for Agent Sandboxing and Interpretability*. In: *Proceedings of the Second Reinforcement Learning Conference*, pp. 2844–2881. arXiv:2504.04608.

[RS59] Michael O. Rabin & Dana Scott (1959): *Finite Automata and Their Decision Problems*. *IBM Journal of Research and Development*.

[Rut00] Jan J. M. M. Rutten (2000): *Universal Coalgebra: A Theory of Systems*. *Theoretical Computer Science* 249(1), pp. 3–80, doi:10.1016/S0304-3975(00)00056-6.

[SBBR10] Alexandra Silva, Filippo Bonchi, Marcello M. Bonsangue & Jan J. M. M. Rutten (2010): *Generalizing the Powerset Construction, Coalgebraically*. In: *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2010)*, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik GmbH, Wadern/Saarbruecken, Germany, p. 12 pages, doi:10.4230/LIPICS.FSTTCS.2010.272.

[SBBR13] Alexandra Silva, Filippo Bonchi, Marcello Bonsangue & Jan J. M. M. Rutten (2013): *Generalizing Determinization from Automata to Coalgebras*. *Logical Methods in Computer Science* Volume 9, Issue 1, p. 1087, doi:10.2168/LMCS-9(1:9)2013.

[SC01] Cosma Rohilla Shalizi & James P. Crutchfield (2001): *Computational Mechanics: Pattern and Prediction, Structure and Simplicity*. *Journal of Statistical Physics* 104(3), pp. 817–879, doi:10.1023/A:1010388907793.

[SSSM22] Jayakumar Subramanian, Amit Sinha, Raihan Seraj & Aditya Mahajan (2022): *Approximate Information State for Approximate Planning and Reinforcement Learning in Partially Observed Systems*. Journal of Machine Learning Research.

[Str65] Charlotte Striebel (1965): *Sufficient Statistics in the Optimum Control of Stochastic Systems*. Journal of Mathematical Analysis and Applications 12(3), pp. 576–592, doi:10.1016/0022-247X(65)90027-2.

[Vir23] Nathaniel Virgo (2023): *Unifilar Machines and the Adjoint Structure of Bayesian Models*. In: Electronic Proceedings in Theoretical Computer Science, 397, pp. 299–317.