Bicategories of Markov Processes

Florence Clerc[1], Harrison Humphrey[1], and Prakash Panangaden[1]

School of Computer Science
McGill University

Abstract. We construct bicategories of Markov processes where the objects are input and output sets, the morphisms (one-cells) are Markov processes and the two-cells are simulations. This builds on the work of Baez, Fong and Pollard, who showed that a certain kind of finite-space continuous-time Markov chain (CTMC) can be viewed as morphisms in a category. This view allows a compositional description of their CTMCs. Our contribution is to develop a notion of simulation between processes and construct a bicategory where the two-cells are simulation morphisms. Our version is for processes that are essentially probabilistic transition systems with discrete time steps and which do not satisfy a detailed balance condition. We have also extended the theory to continuous space processes.

1 Introduction

A recent paper by Baez, Fong and Pollard [1] develops a compositional framework for Markov processes. More precisely, they work with finite-state processes with a population associated with each state. Transitions are governed by *rates* and are memoryless. Thus, they are working with continuous-time Markov chains (see *e.g.* [8]). The important innovation in their work is to define "open" Markov chains with inputs and outputs. This allows them to connect Markov chains together and build more complex ones from simpler ones.

Our work is inspired by their treatment but differs in two significant ways. First, we work with Markov processes viewed operationally. That is, the states represent states of a transition system and the system moves between states according to a probabilistic law: thus they are closer in spirit to probabilistic automata. We do not impose a detailed balance condition; it would not make any sense in the scenario we are examining. Importantly we allow continuous state spaces; which forces us into some measure-theoretic considerations. The crucial idea that we borrow from Baez et al. [1] is the use of *open* processes that can be composed. Though the details are different from [1] essentially the mathematics is inspired by their work and the work of Fong [3] on decorated cospans.

The second significant difference is the development of a bicategorical picture. The idea here is to have two-cells that capture *simulation*. The concepts of simulation and bismulation have played a central role in the development of

process algebra [5, 6, 10] and the probabilistic version has been similarly important [4, 9]. We have used simulation morphisms similar in spirit to those used by Desharnais et al. [2, 9].

Dedication

It is a pleasure for the third author to dedicate this paper to Kim Larsen. Kim's fundamental work on probabilistic bisimulation nearly 30 years ago was a breakthrough and the inspiration for his own work [2] on the subject. Ever since then he has been infected with the probability bug and has maintained ties with Kim and his research group. This paper also deals with exactly those topics and we hope that Kim will accept this as a tribute to his remarkable career and achievements.

2 Discrete Markov Processes

We begin by developing the theory on finite state spaces so that we can postpone the measure theory issues until later. It is pleasing that the measure theory and the category theory can be more or less "factored" into separate sections.

Definition 1 *Given a finite set M, a* Markov kernel *on M is a map $\tau : M \times M \to [0, 1]$ such that for all $m \in M$, $\tau(m, .)$ is a subprobability measure on M. A* labelled Markov process *on M is a collection (τ_a) of Markov kernels on M that is indexed by a set of actions Act.*

Markov processes are the standard model of memoryless probabilistic dynamical systems like a probabilistic program executing or particles moving over time subject to random influences. Let us fix a set of actions *Act* throughout this paper. These actions correspond to interactions between the process and the environment; for instance, a user performing control actions on a stochastic system.

Note that here we are only requiring subprobability measures. This is because it might be the case that the process does not terminate and some of the probability mass might be lost. We also want to have some cases where the transition probabilities are zero which subprobability distributions allow us to accommodate.

As in [1], we can view our labelled Markov processes as morphisms between input and output sets.

Definition 2 *Given two finite sets X, Y, a* discrete labelled Markov process *(DLMP) from X to Y is a tuple $(M, (\tau_a)_{a \in Act}, i, o)$ consisting of a finite set M, a labelled Markov process $(\tau_a)_{a \in Act}$ on M, and two injective morphisms $i : X \to M$ and $o : Y \to M$ called* input *and* output.

We also require that for $a \in Act$, $y \in Y$ and $m \in M$, $\tau_a(o(y), m) = 0$.

The last condition says that when the process reaches a state corresponding to the output it stops there. When we compose processes, these will become inputs to the next process and will be subject to a new dynamics. Note that a state can be input and output: this means that if the system is started in this state it will just stay there. We will also write $\tau_a(m, A)$, where $A \subseteq M$, to mean $\sum_{x \in A} \tau_a(m, x)$.

The key difference between the standard definition of finite labelled Markov process and this definition of DLMP is the use of input and output sets that allows us to specify the state in which the system is at the start and the state when the experiment stops.

An outside observer is allowed to influence the system using the actions in Act, which result in a probabilistic response by the system; the response to performing the action a at state m is given by the final state (sub)distribution $\tau_a(m, \cdot)$. Particles flow through the Markov process, beginning at inputs, according to the kernels τ_a, until they reach an output state. When a system hits an output state it stops. Later we will describe how composed systems behave; essentially the output states become the input states of the next system.

Let us illustrate this definition using the example of a pinball machine. The position of the ball represents the state of the process. The ball is introduced when the player starts the game; this is the input state. The ball then moves around (this is the process) with the player using flippers (actions) to act on its trajectory. The game ends when the ball reaches the drain (output).

Note that the requirement on the Markov kernels is not symmetric between inputs and outputs. This is a direct consequence of the fact that input and output correspond respectively to start and end of observation or experiment. In that setting, a start state can lead to another start state whereas once the experiment is over, it cannot evolve anymore.

2.1 Viewing DLMPs as morphisms

Viewing Markov processes as processes from inputs to outputs makes it tempting to construct a category **DLMP**. However, we will see that there is a problem with the composition being associative only up to isomorphism. The objects are finite sets and the morphisms $X \to Y$ are DLMPs from X to Y.

Let us first give an intuition for this composition: this corresponds to cascading the Markov processes one after the other by identifying states that were outputs in the first DLMP with inputs in the second DLMP. Consider three finite sets X, Y, Z and two DLMPs

$$\mathcal{M} := (M, (\tau_a^M)_{a \in Act}, i_M, o_M) : X \to Y$$

and

$$\mathcal{N} := (N, (\tau_a^N)_{a \in Act}, i_N, o_N) : Y \to Z$$

The category of finite sets and functions between them has pushouts. Let us denote $M +_Y N$ the pushout of M and N along i_N and o_M, and let j_N and j_M be the inclusion maps.

$$\begin{array}{ccc} Y & \xrightarrow{o_M} & M \\ {\scriptstyle i_N}\downarrow & & \downarrow{\scriptstyle j_M} \\ N & \xrightarrow[j_N]{} & M +_Y N \end{array}$$

The pushout $M +_Y N$ can be expressed as $M +_Y N := (M \uplus N)/\sim$ where $\sim$ denotes the smallest equivalence relation on $M + N$ such that for all $y \in Y$, $j_M(o_M(y)) \sim j_N(i_N(y))$.

The composition of $\mathcal{M}$ and $\mathcal{N}$ denoted $\mathcal{N} * \mathcal{M}$ is the DLMP with input X and output Z defined as follows.

$$\mathcal{N} * \mathcal{M} := (M +_Y N, (\tau'_a)_{a \in Act}, j_M \circ i_M, j_N \circ o_N)$$

where, for $m, n \in M +_Y N$

$$\tau'_a(m,n) = \begin{cases} \tau_a^N(m,n) \text{ if } m,n \in j_N(N) \\ \tau_a^M(m,n) \text{ if } m,n \notin j_N(N) \text{ and } m,n \in j_M(M) \\ 0 \text{ otherwise} \end{cases}$$

Note that if m and n are both outputs of the first DLMP and inputs of the second one, we use τ^N.

The universal property of the pushout in **FinSet** ensures that composition is associative only up to isomorphism. This will be explained in more detail in the coming section; but note that it prevents us from constructing a category of DLMPs. Given any finite set X, the identity 1_X is the DLMP $(X, (\tau_a)_{a \in Act}, id_X, id_X)$, where for all $a \in$ Act, and for all $x, y \in X$, $\tau_a(x,y) = 0$. Note that it is only an identity up to isomorphism.

2.2 Simulations as Morphisms between DLMPs

Given two Markov processes with the same input and output sets, it is natural to ask whether they are related in some way or not. To this end, we first introduce the notion of simulation, and then show how it provides a natural framework for extending the previous construction to a bicategory.

Definition 3 *Given two DLMPs $\mathcal{N} = (N, (\tau_a^N)_{a \in Act}, i_N, o_N)$ and $\mathcal{M} = (M, (\tau_a^M)_{a \in Act}, i_M, o_M)$ defined with the same input and output sets, a* simulation *of $\mathcal{N}$ by $\mathcal{M}$ is a function $f : N \to M$ on the state spaces satisfying the following conditions:*

- *$f \circ i_N = i_M$ and $f \circ o_N = o_M$, and*
- *for all $a \in Act$, $n \in N$ and $m \in M$, $\tau_a^M(f(n), m) \geq \tau_a^N(n, f^{-1}(m))$.*

where $f^{-1}(m)$ stands for $f^{-1}(\{m\})$. In such a case, we say that $\mathcal{M}$ simulates *$\mathcal{N}$ and write $f : \mathcal{N} \Rightarrow \mathcal{M}$.*

Given two finite sets X and Y, we have the "hom-set" $\mathbf{DLMP}(X, Y)$ of the previously defined "category **DLMP**." The quotation marks signify that we don't really have a category. However, it is possible to extend the set $\mathbf{DLMP}(X, Y)$ to a category with objects the DLMPs from X to Y and as morphisms simulations between such DLMPs in such a way that we obtain a bicategory. We carry this out in the next subsection.

The composition of two simulations with the same input and output sets is given by standard function composition; it is denoted $\circ$. The standard composition is associative which ensures that $\circ$ is also associative.

Proof. Let us now check that the composition of two simulations is a simulation. Consider two simulations $f : \mathcal{M}_1 \Rightarrow \mathcal{M}_2$ and $g : \mathcal{M}_2 \to \mathcal{M}_3$ with $\mathcal{M}_k = (M_k, (\tau_a^k)_{a\in Act}, i_k, o_k) : X \to Y$. Note that for any m in M_1 and n in M_3:

$$\tau_a^3(g \circ f(m), n) \geq \tau_a^2(f(m), g^{-1}(n)) \geq \tau_a^1(m, (g \circ f)^{-1}(n))$$

using the fact that g and f are both simulations. Finally note that $(g \circ f) \circ i_1 = g \circ i_2 = i_3$ and similarly for the output map. This proves that the composition of two simulations is a simulation.

Given a DLMP $\mathcal{M} = (M, (\tau_a^M)_{a\in Act}, i_M, o_M)$, the identity $\mathrm{id}_{\mathcal{M}}$ is the identity on the underlying set id_M. It is indeed an identity for the composition we have just defined.

2.3 The Bicategory DLMP

We had started with trying to construct a category **DLMP** with finite sets as objects and DLMPs as morphisms. We have constructed a categorical structure on the hom-set $\mathbf{DLMP}(X, Y)$ for all finite sets X, Y. It is natural to further extend it in order to make **DLMP** into a bicategory.

One of the things missing from our construction is a horizontal composition, namely for every triple of finite sets X, Y and Z a functor

$$c_{XYZ} : \mathbf{DLMP}(Y, Z) \times \mathbf{DLMP}(X, Y) \to \mathbf{DLMP}(X, Z)$$

Given two DLMPs $\mathcal{M} : X \to Y$ and $\mathcal{N} : Y \to Z$, $c_{XYZ}(\mathcal{N}, \mathcal{M})$ is their composition $\mathcal{N} * \mathcal{M}$ defined in Section 2.1.

Let us now define the functor c_{XYZ} acting on the simulations. Let us consider four DLMPs (with $k = 1, 2$) :

$$\mathcal{M}_k = (M_k, (\tau_a^{M,k})_{a\in Act}, i_{M,k}, o_{M,k}) : X \to Y$$

and

$$\mathcal{N}_k = (N_k, (\tau_a^{N,k})_{a\in Act}, i_{N,k}, o_{N,k}) : Y \to Z$$

as well as two simulations

$$f : \mathcal{M}_1 \Rightarrow \mathcal{M}_2 \quad \text{and} \quad g : \mathcal{N}_1 \Rightarrow \mathcal{N}_2$$

Let us denote $j_{N,k} : N_k \to M_k +_Y N_k$ and $j_{M,k} : M_k \to M_k +_Y N_k$ the pushout maps obtained by performing the horizontal composition $\mathcal{N}_k * \mathcal{M}_k$.

We are now ready to define their horizontal composition $c_{XYZ}(g, f) : \mathcal{N}_1 * \mathcal{M}_1 \Rightarrow \mathcal{N}_2 * \mathcal{M}_2$ as follows. For $m \in M_1 +_Y N_1$,

$$(g * f)(m) = \begin{cases} j_{N,2} \circ g(n') \text{ if } \exists n' \in N_1 \text{ such that } m = j_{N,1}(n') \\ j_{M,2} \circ f(m') \text{ if } \exists m' \in M_1 \text{ such that } m = j_{M,1}(m') \end{cases}$$

We denote $c_{XYZ}(g, f)$ by $g * f$.

Note that $g * f(m)$ is well defined.

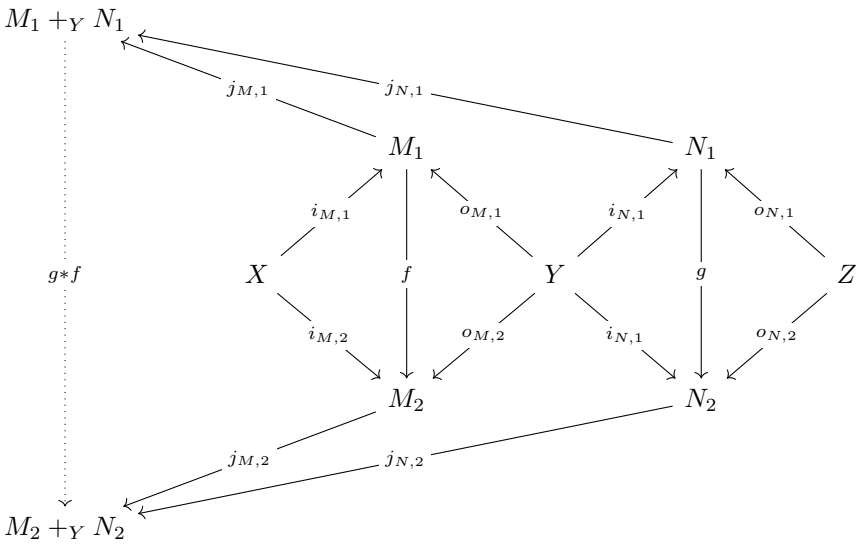
Proof. Assume that there exists n' in N_1 and $m' \in M_1$ such that $m = j_{N,1}(n') = j_{M,1}(m')$. By definition of the pushout, there exists y in Y such that $m' = o_{M,1}(y)$ and $n' = i_{N_1}(y)$.

$$\begin{aligned} (j_{M,2} \circ f)(m') &= (j_{M,2} \circ f \circ o_{M,1})(y) \\ &= (j_{M,2} \circ o_{M,2})(y) \quad \text{as } f \text{ is a simulation} \\ &= (j_{N,2} \circ i_{N,2})(y) \quad \text{using the pushout} \\ &= (j_{N,2} \circ g \circ i_{N,1})(y) \quad \text{as } g \text{ is a simulation} \\ &= (j_{N,2} \circ g)(n') \end{aligned}$$

The case where there would be n_1 and n_2 in N_1 (resp. m_1 and m_2 in M_1) satisfying both the first (resp. second) condition is prevented by the injectivity of $i_{N,1}$ (resp. $o_{M,1}$).

Lemma 1. *The horizontal composition $g * f$ is a simulation.*

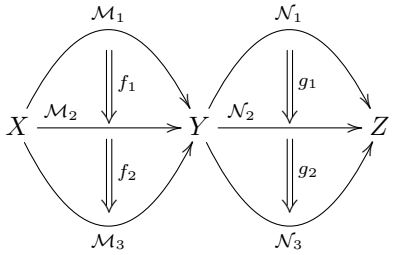
Proof. Diagrammatically, the situation is the following :



In order to prove that it is indeed a simulation, we have first to prove that $(g*f)\circ j_{M,1}\circ i_{M,1} = j_{M,2}\circ i_{M,2}$. Let x in X, note that $i_{M,1}(x) \in M_1$, therefore by definition of $g*f$, $(g*f)\circ j_{M,1}\circ i_{M,1}(x) = j_{M,2}\circ f(i_{M,1}(x))$. But f is a simulation, hence $f(i_{M,1}(x)) = i_{M,2}(x)$ proving the desired equality. The corresponding equality with output maps is proven similarly.

Let us denote $(\tau_a^k)_{a\in Act}$ the Markov process corresponding to the composition $\mathcal{N}_k * \mathcal{M}_k$. There remains to prove that for all $a \in Act$, $m_1 \in M_1 +_Y N_1$ and $m_2 \in M_2 +_Y N_2$, $\tau_a^2((g*f)(m_1), m_2) \geq \tau_a^1(m_1, (g*f)^{-1}(m_2))$. There are many cases that correspond to the different cases for $g*f$, τ_a^1 and τ_a^2. The proof is straightforward but tedious.

Lemma 2. *The exchange law holds. Namely, let $\mathcal{M}_k, \mathcal{N}_k$ with $k = 1, 2, 3$ be DLMPs with $\mathcal{M}_k : X \to Y$ and $\mathcal{N}_k : Y \to Z$ and let us consider simulations $f_1 : \mathcal{M}_1 \Rightarrow \mathcal{M}_2$, $f_2 : \mathcal{M}_2 \Rightarrow \mathcal{M}_3$, $g_1 : \mathcal{N}_1 \Rightarrow \mathcal{N}_2$ and $g_2 : \mathcal{N}_2 \Rightarrow \mathcal{N}_3$ corresponding to*

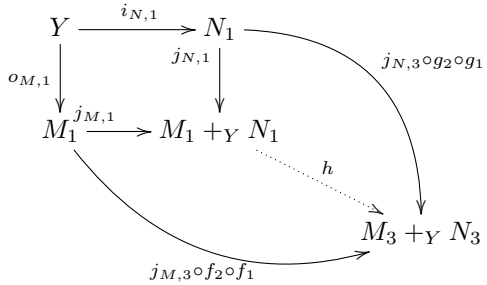


then $(g_2 \circ g_1) * (f_2 \circ f_1) = (g_2 * f_2) \circ (g_1 * f_1)$.

Proof. Let us denote as usual $j_{M,k} : M_k \to M_k +_Y N_k$ and $j_{N,k} : N_k \to M_k +_Y N_k$ for $k = 1, 2, 3$ the corresponding pushout maps. As g_1 and g_2 are simulations, we know that

$$j_{N,3} \circ g_2 \circ g_1 \circ i_{N,1} = j_{N,3} \circ g_2 \circ i_{N,2} = j_{N,3} \circ i_{N,3}$$

and similarly $j_{M,3} \circ f_2 \circ f_1 \circ o_{M,1} = j_{M,3} \circ o_{M,3}$. By the universal property of the pushout, there is a unique map h making the following diagram commute:



It can be easily verified that both $(g_2 \circ g_1) * (f_2 \circ f_1)$ and $(g_2 * f_2) \circ (g_1 * f_1)$ work for h, hence they are equal.

Lemma 3. *The horizontal composition is associative up to isomorphisms, i.e. for any finite sets X, Y, Z and W, we have natural isomorphisms called the associators*

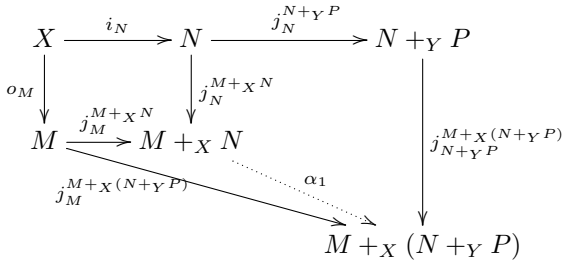
$$\alpha_{WXYZ} : c_{WYZ} \circ (id, c_{WXY}) \to c_{WXZ} \circ (c_{XYZ}, id)$$

Proof. Let us consider three DLMPs $\mathcal{M} = (M, (\tau_a^M), i_M, o_M) : W \to X$, $\mathcal{N} = (N, (\tau_a^N), i_N, o_N) : X \to Y$ and $\mathcal{P} = (P, (\tau_a^P), i_P, o_P) : Y \to Z$. Let us construct the associator $\alpha_{\mathcal{MNP}} : \mathcal{P} * (\mathcal{N} * \mathcal{M}) \Rightarrow (\mathcal{P} * \mathcal{N}) * \mathcal{M}$, i.e. a simulation map

$$\alpha_{\mathcal{MNP}} : (M +_X N) +_Y P \to M +_X (N +_Y P)$$

We will denote the pushout maps $j_M^{M+_Y N} : M \to M +_Y N$ etc.

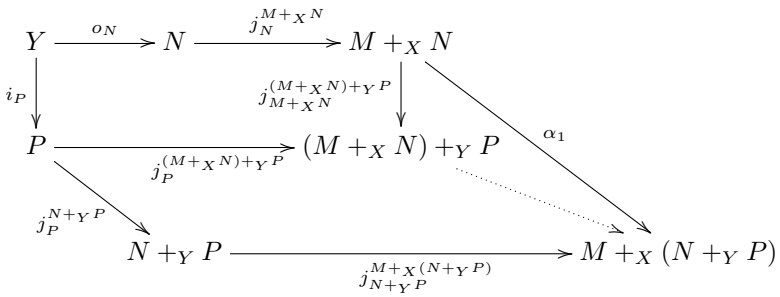
First note that $X \xrightarrow{i_N} N \xrightarrow{j_N^{N+_Y P}} N +_Y P$ is the input map of the DLMP $\mathcal{N} * \mathcal{P}$, making the outer diagram commute:



By the universal property of the pushout $M +_X (N +_Y P)$, there exists a unique map $\alpha_1 : M +_X N \to M +_X (N +_Y P)$ making the above diagram commute.

To show that the outer diagram commutes, we calculate as follows:

$$\begin{aligned}\alpha_1 \circ j_N^{M+_Y N} \circ o_N &= j_{N+_Y P}^{M+_X(N+_Y P)} \circ j_N^{N+_Y P} \circ o_N \quad \text{using the definition of } \alpha_1 \\ &= j_{N+_Y P}^{M+_X(N+_Y P)} \circ j_P^{N+_Y P} \circ i_P \quad \text{using the pushout square of } N +_Y P\end{aligned}$$



By universal property of the pushout $(M +_X N) +_Y P$, there exists a unique map $(M +_X N) +_Y P \to M +_X (N +_Y P)$ making this diagram commute. We call this map $\alpha_{\mathcal{MNP}}$. Note that we could have constructed the associator from the explicit definition of the pushout given in Section 2.1.

Naturality and isomorphism of the associator follow from similar constructions and the fact that all pushout maps are injective as the input and output maps are injective.

Remember that we had defined identity DLMP $1_X = (X, (0)_{a\in Act}, id_X, id_X)$. Similar constructions using pushouts give us two natural isomorphisms corresponding to the unitors : for all $\mathcal{M} : X \to Y$ a DLMP, we have

$$\lambda_{\mathcal{M}} : \mathcal{M} * 1_X \to \mathcal{M} \text{ and } \rho_{\mathcal{M}} : \mathcal{M} \to 1_Y * \mathcal{M}$$

Pentagon identities and triangle identities are proven using similar computations. This proves the following result: the main goal of this section.

Theorem 4 DLMP *is a bicategory.*

3 Continuous State Space

While the finite case is interesting to start with, in many cases of interest the underlying state space of an LMP is not finite but an arbitrary measurable set or perhaps a more restricted structure like a Polish space or an analytic space. However, most of the work we did in the previous section does not rely on LMPs having a finite state space and it becomes very tempting to extend the bicategory **DLMP** we just constructed to a more general notion of LMP. It is not as straightforward as it may seem as the output map is more complicated in the continuous case. The restriction to analytic spaces is important for proving the logical characterization of bisimulation or simulation. Since we are not doing that here we will consider general measurable spaces.

3.1 LMP and simulation in the continuous case

Definition 5 *Given a measurable space (M, Σ) a* Markov kernel *is a function $\tau : M \times \Sigma \to [0, 1]$ where for each $m \in M$ the function $\tau(m, \cdot)$ is a subprobability measure on (M, Σ) and for each measurable set $B \in \Sigma$ the function $\tau(\cdot, B) : M \to [0, 1]$ is measurable where $[0, 1]$ is equipped with the standard Borel-algebra. A* labelled Markov process *is a collection (τ_a) of Markov kernels on (M, Σ) that is indexed by a set of actions Act.*

Let us now extend our previous definition of DLMPs to deal with the continuous case.

Definition 6 *Given two finite sets X and Y, a* continuous labelled Markov process *(CLMP) from X to Y is a tuple $(M, \Sigma, (\tau_a)_{a\in Act}, i, o)$ consisting of (M, Σ) a measurable space, a labelled Markov Process $(\tau_a)_{a\in Act}$, an injective function $i : X \to M$ and a function $o : Y \to \Sigma$ such that for all y_1 and y_2 in Y $o(y_1) \cap o(y_2) = \emptyset$, satisfying the following additional condition for all $a \in A$:*

$$\text{for all } y \in Y,\ m \in o(y) \text{ and } B \in \Sigma \quad \tau_a(m, B) = 0$$

Note that here we have an input point but a (measurable) output *set*. To avoid painfully long notations, we will also write $o(Y)$ for the set $\bigcup_{y\in Y} o(y) \in \Sigma$.

We now adapt the definition of simulation to this setting.

Definition 7 *Given two CLMPs* $\mathcal{N} = (N, \Lambda, (\tau_a^N)_{a \in Act}, i_N, o_N)$ *and* $\mathcal{M} = (M, \Sigma, (\tau_a^M)_{a \in Act}, i_M, o_M)$ *defined with the same input and output sets, a* simulation *of* $\mathcal{N}$ *by* $\mathcal{M}$ *is a measurable function* $f : N \to M$ *on the state spaces satisfying the following conditions:*

- $f \circ i_N = i_M$ *and* $o_N = f^{-1} \circ o_M$, *and*
- *for all* $a \in Act$, $n \in N$ *and* $B \in \Sigma$, $\tau_a^M(f(n), B) \geq \tau_a^N(n, f^{-1}(B))$.

In such a case, we say that $\mathcal{M}$ simulates $\mathcal{N}$ *and write* $f : \mathcal{N} \Rightarrow \mathcal{M}$.

3.2 The bicategory CLMP

We now extend what was done in the finite case to the continuous case in order to construct the bicategory **CLMP**.

Given two finite sets X, Y, there is a category $\mathbf{CLMP}(X, Y)$ which has as objects the CLMPs $X \to Y$ and as morphisms the simulations between them. Composition is given by the standard composition on their underlying sets and the identities are the standard identities on the underlying state spaces.

The next order of business is to define the horizontal composition both on the CLMPs and the simulations. Let us start with the CLMPs.

Given three finite sets X, Y and Z and two CLMPs $\mathcal{M} = (M, \Sigma, i_M, o_M, \tau^M) : X \to Y$ and $\mathcal{N} = (N, \Lambda, i_N, o_N, \tau^N) : Y \to Z$, there are two inclusion maps $j_N : N \to M + N$ and $j_M : M \to M + N$. We then define the relation $\sim$ on $M + N$ as the smallest equivalence such that

$$\forall y \in Y \ \forall m \in o_M(y) \ j_M(m) \sim j_N(i_N(y))$$

We then define the quotient map q between measurable spaces $q : (M + N, \Sigma + \Lambda) \to ((M + N)/\sim, (\Sigma + \Lambda)/\sim)$ where $(\Sigma + \Lambda)/\sim$ is the smallest σ-algebra such that q is measurable.

Note that here we are mimicking the explicit construction of the pushout given in the finite case. We will therefore also denote $(N + M)/\sim$ as $N +_Y M$ and $(\Sigma + \Lambda)/\sim$ as $\Sigma +_Y \Lambda$. We define the horizontal composition of $\mathcal{M}$ and $\mathcal{N}$ as:

$$\mathcal{N} * \mathcal{M} = (M +_Y N, \Sigma +_Y \Lambda, q \circ j_M \circ i_M, q \circ j_N \circ o_N, \tau')$$

where the LMP is defined for $m \in M +_Y N$ and $B \in \Sigma +_Y \Lambda$ as

$$\tau'_a(m, B) = \begin{cases} \tau_a^M(m', j_M^{-1} q^{-1}(B)) \text{ if } \exists m' \in M \setminus o_M(Y) \ m = q \circ j_M(m') \\ \tau_a^N(n', j_N^{-1} q^{-1}(B)) \text{ if } \exists n' \in N \ m = q \circ j_M(n') \\ 0 \text{ otherwise} \end{cases}$$

Note here how the condition on the input and output maps is used : remember that the input map is injective and that the output maps gives sets that are pairwise disjoint. This ensures that if $m_1 \sim m_2$ with m_1 and m_2 in M then there exists y in Y such that m_1 and m_2 are in $o_M(y)$ and if $n_1 \sim n_2$ with n_1 and n_2 in N then $n_1 = n_2$. This guarantees that τ'_a is well-defined.

The identity is the same as the one we have defined in the discrete case : let X be a finite set and let Σ be the discrete σ-algebra on X, then the identity is

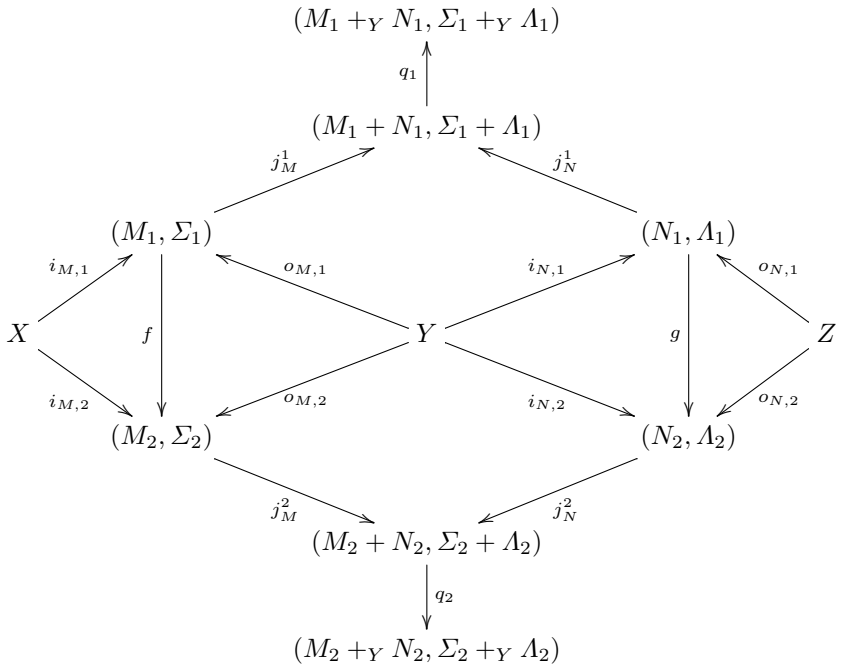
$$1_X = (X, \Sigma, (\tau_a), \mathrm{id}_X, o_X)$$

where $\tau_a(x, B) = 0$ for all $x \in X$ and $B \in \Sigma$ and $o_X(x) = \{x\}$.

For every triple of finite sets X, Y and Z, we define the horizontal composition on the simulations. Consider $f : \mathcal{M}_1 \Rightarrow \mathcal{M}_2 : X \to Y$ and $g : \mathcal{N}_1 \Rightarrow \mathcal{N}_2 : Y \to Z$ where $\mathcal{M}_k = (M_k, \Sigma_k, \tau^{M,k}, i_{M,k}, o_{M,k})$ and $\mathcal{N}_k = (N_k, \Lambda_k, \tau^{N_k}, i_{N,k}, o_{N,k})$ $(k = 1, 2)$. We use similar notations as for the composition of CLMPs but index them by 1 or 2 (see following diagram).

We define their horizontal composition as

$$\begin{aligned} g * f : M_1 +_Y N_1 &\to M_2 +_Y N_2 \\ n &\mapsto q_2 \circ j^2_N \circ g(n') \text{ if } \exists n' \in N_1 \ n = q_1 \circ j^1_N(n') \\ m &\mapsto q_2 \circ j^2_M \circ f(m') \text{ if } \exists m' \in M_1 \ m = q_1 \circ j^1_M(m') \end{aligned}$$



This is again mimicking what happens in the finite case. Note that the remark used previously to show that the horizontal composition of DLMPs is well-defined is used here to prove that the horizontal composition of the CLMPs is well-defined. The proofs with the associators and the unitors are similar to

the finite case except that they rely on the universal property of the quotient instead of the universal property of the pushout.

We can now state the main result of this paper.

Theorem 8 CLMP *is a bicategory.*

4 Conclusions

We have developed a notion of bicategory of Markov processes where the two-cells capture the notion of simulation. The original paper of Baez, Fong and Pollard developed a compositional theory of a certain class of CTMCs. We have developed an analogous theory for Markov processes in both discrete and continuous state-space versions. By adding the two-cells we have incorporated one of the most powerful and widely used tools for reasoning about the behaviour of Markov processes and this opens the way for compositional reasoning.

Of course, this paper is just a start. There are many interesting directions to explore. Perhaps the most pressing is to understand how feedback can be incorporated via a trace structure. Certain categories of probabilistic relations do have a traced monoidal structure; it remains to be seen how to incorporate that here in a manner consistent with the two-cell structure. We are also working on using more general coalgebras as the morphisms instead of just Markov processes.

In earlier work [9] logical formalisms (modal logics) for reasoning about bisimulation have been developed. Here we have the framework where one can think about compositional logical reasoning. In a paper about a decade ago Mislove et al. [7] have studied duality for Markov processes (our CLMPs) and also developed a notion of composing Markov processes. We have not yet worked out the relations between that framework and ours but clearly it is an interesting topic to be examined.

Acknowledgements

We are very grateful to Brendan Fong for helpful discussions. We thank the reviewers for their detailed comments and feedback. This research has been supported by a research grant from NSERC.

References

1. Baez, J.C., Fong, B., Pollard, B.S.: A compositional framework for Markov processes (Sept 2015), arXiv:1508.06448
2. Desharnais, J., Edalat, A., Panangaden, P.: Bisimulation for labeled Markov processes. Information and Computation 179(2), 163–193 (Dec 2002)
3. Fong, B.: Decorated cospans. Theory and Applications of Categories 30, 1096–1120 (2015)
4. Larsen, K.G., Skou, A.: Bisimulation through probablistic testing. Information and Computation 94, 1–28 (1991)

5. Milner, R.: A Calculus for Communicating Systems, Lecture Notes in Computer Science, vol. 92. Springer-Verlag (1980)
6. Milner, R.: Communication and Concurrency. Prentice-Hall (1989)
7. Mislove, M., Ouaknine, J., Pavlovic, D., Worrell, J.: Duality for labelled Markov processes. In: Walukiewicz, I. (ed.) Foundations of Software Science and Computation Structures, FOSSACS. Lecture Notes In Computer Science, vol. 2987, pp. 393–407 (2004)
8. Norris, J.R.: Markov Chains. Cambridge Series in Statistical and Probabilistic Mathematics, Cambridge University Press (1997)
9. Panangaden, P.: Labelled Markov Processes. Imperial College Press (2009)
10. Park, D.: Concurrency and automata on infinite sequences. In: Proceedings of the 5th GI Conference on Theoretical Computer Science, pp. 167–183. No. 104 in Lecture Notes In Computer Science, Springer-Verlag (1981)