

# A Bicategorical Approach to Transition Systems

Harrison Humphrey

School of Computer Science

McGill University, Montreal

June, 2017

A thesis submitted to McGill University in partial fulfillment of the  
requirements of the degree of Master of Science.

©Harrison Humphrey, 2017



# Contents

<b>Abstract</b>	<b>1</b>
<b>Acknowledgements and Contribution of Authors</b>	<b>2</b>
<b>Introduction</b>	<b>3</b>
<b>1 Background Material</b>	<b>5</b>
1.1 Composing Markov Processes . . . . .	5
1.2 Bicategories . . . . .	8
<b>2 Labelled Transition Systems</b>	<b>11</b>
2.1 Viewing LTSs as morphisms . . . . .	12
2.2 Simulations as Morphisms between LTSs . . . . .	14
2.3 The Bicategory <b>LTS</b> . . . . .	16
<b>3 Markov Processes</b>	<b>21</b>

3.1	Discrete Markov Processes . . . . .	21
3.1.1	Viewing DLMPs as morphisms . . . . .	23
3.1.2	Simulations between DLMPs . . . . .	24
3.1.3	The Bicategory <b>DLMP</b> . . . . .	26
3.2	Continuous State Space . . . . .	31
3.2.1	LMP and simulation in the continuous case . . . . .	32
3.2.2	The bicategory <b>CLMP</b> . . . . .	33
4	<b>Conclusion</b>	<b>37</b>

# Abstract

The goal of this thesis is to show that a wide range of operational systems can be reasoned about *compositionally*. The formal tool we use to explore this is the notion of a *bicategory*. Indeed, we construct bicategories for Markov processes and transition systems, where the objects are input and output sets, the morphisms (one-cells) are the 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 picture allows a compositional description of their CTMCs. Our contribution is to develop a notion of simulation for both labelled transition systems and Markov processes which allows us to lift this framework to the 2-categorical level (where the two-cells are simulation morphisms).

Cette thèse a pour but de démontrer qu’une série de systèmes opérationnels peuvent mieux être compris de manière compositionnelle. Pour effectuer ceci, on utilise la notion de *bicatégorie*. Notamment, cette thèse construit une bicatégorie de *Markov processes* et de *transition systems*, où les objets sont des ensembles, les “one-cells” sont les systèmes opérationnels en question, et les “two-cells” sont des *simulations*. Ce traitement est inspiré par les idées de Baez, Fong et Pollard, qui ont montré que certains “continuous-time Markov chains” peuvent être compris comme des morphismes dans une certaine catégorie. Notre contribution est de développer une notion de simulation pour les “labelled transitions systems” ainsi que les “continuous-time Markov chains” qui permet un raisonnement au niveau de 2-catégorie.

## Acknowledgements

I wish to express my sincerest gratitude to my supervisor, Professor Prakash Panangaden, for guiding me through my time as a student at McGill University. He has always been available and able to patiently provide guidance and insight, in matters both in and out of academia. I am eternally grateful to him for introducing me to the world of theoretical computer science.

I also wish to thank the other members of the RL lab at McGill, especially Florence, Ira, Nicolas, and Matt for creating such a hospitable research environment.

I gratefully acknowledge the financial support received from the Natural Sciences and Engineering Research Council of Canada (NSERC), and McGill University.

Finally, I would like to thank my family for their invaluable support.

## Contribution of Authors

The work detailed in this thesis was undertaken by myself (Harrison Humphrey), my supervisor Prakash Panagaden, and my fellow student and research partner Florence Clerc. It will appear in [1].

# Introduction

A recent paper by Baez, Fong and Pollard [2] 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.* [3]). 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.

The work presented in this thesis is inspired by their treatment but differs in two significant ways. First, we work with a wider set of processes: we look at Markov processes (with both discrete and continuous state spaces), as well as labelled transition systems. Second, we view them *operationally*. That is, the states represent states of a transition system and the system moves between states according to a non-deterministic or probabilistic law: thus they are closer in spirit to 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. [2] is the use of *open* processes that can be composed, using the categorical notion of pushout. Though the details are different from [2], the mathematics is inspired by their work and the work of Fong [4] on decorated cospans. The second significant difference is the development of a bicategorical picture. The idea here is to have two-cells that capture *simulations*. The concepts of simulation and bismulation have played a central role in the development of process algebra [5, 6, 7] and the probabilistic version has been similarly important [8, 9]. We have used simulation morphisms similar in spirit to those used by Desharnais et al. [10, 9].

Our goal will be to present the prerequisite knowledge in chapter 1, before tackling labelled transition systems in chapter 2 and Markov processes in chapter 3. Our capstone results will be to show that we can indeed construct bicategories where transition systems are the 0-cells (Theorem 14 of Section 2) and where Markov processes are the 0-cells (Theorem 18 of Section 3).



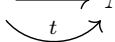
# Chapter 1

## Background Material

### 1.1 Composing Markov Processes

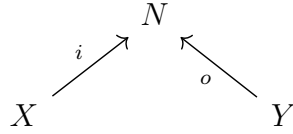
The content of this thesis is inspired by a recent paper [2] by Baez, Fong, and Pollard which looks at a compositional framework for Markov chains. In [2] a continuous-time Markov chain is seen as a way to specify the dynamics of a population which is spread across some finite set of states. More precisely, they look at *open* Markov processes. In these, the population is allowed to flow in or out of certain designated input and output states, or ‘terminals’. If the outputs of one open system match the inputs of another, it is explained how to glue them together, or ‘compose’ them, and obtain a new open system. This makes the Markov processes into morphisms of a certain category.

Indeed, a Markov process is defined as a diagram:

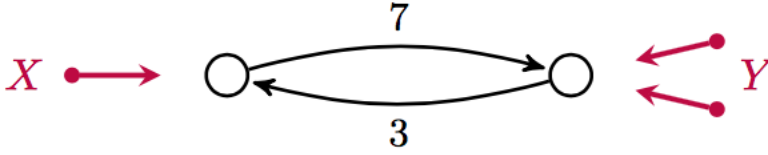
$$(0, \infty) \xleftarrow{r} E \xrightarrow{s} N$$


Here,  $N$  is a finite set of nodes or states,  $E$  is a finite set of edges,  $s, t : E \rightarrow N$  assign to each edge its source and target, and  $r : E \rightarrow (0, +\infty)$  assigns a rate constant to each edge  $e \in E$ . In this situation we call  $M$  a Markov process on  $N$ . If  $e \in E$  has source  $i$  and target  $j$ , we write  $e : i \rightarrow j$ .

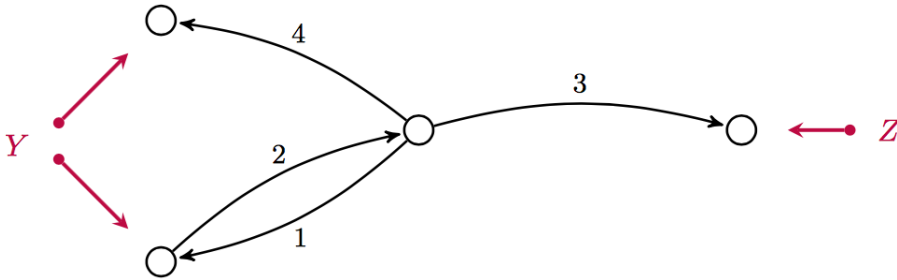
To give these processes a directionality, the paper [2] uses decorated cospans. Following the paper's example, if we have the following diagram:



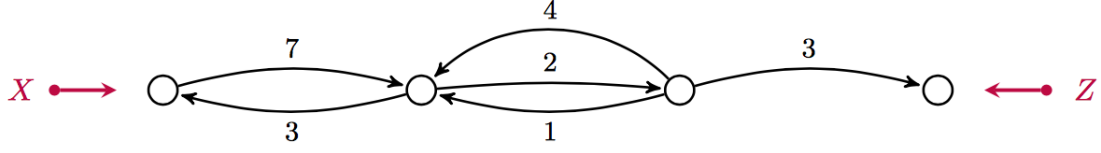
together with a Markov process on  $N$  (as defined above), then we say that  $M$  is an open Markov process from  $X$  to  $Y$ , and we write  $M : X \rightarrow Y$ . If we have such a process



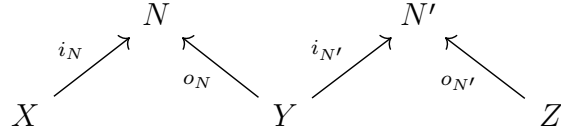
As well as an open Markov process  $M' : Y \rightarrow Z$  as follows:



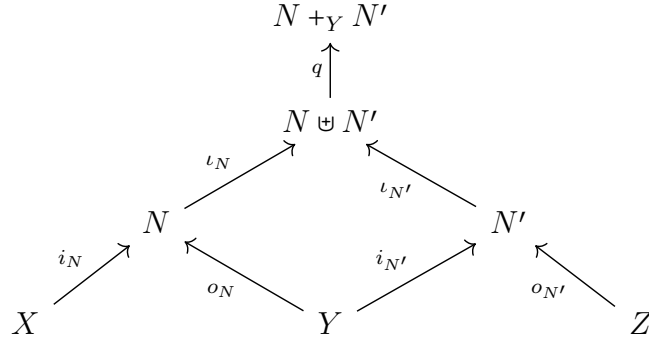
Then intuitively, we should be able to “glue” them together, and their composition should look like the following:



The use of decorated cospans [2] allows one to formalize this process of composition. Putting two cospans side by side as follows, (effectively composing them)

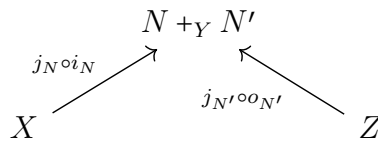


yields a new cospan by looking at the following picture:

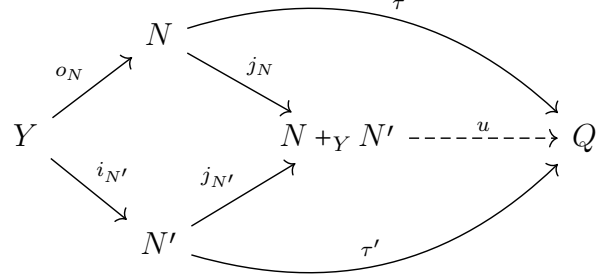


In the above,  $\iota$  represents the obvious inclusions, and  $q$  is a quotient map taking  $N \uplus N'$  to  $N +_Y N' / \sim$  where  $\sim$  is the smallest equivalence relation on  $N \uplus N'$  such that for all  $y \in Y$ ,  $\iota_N(o_N(y)) \sim \iota_{N'}(i_{N'}(y))$ . Note that  $N +_Y N'$  is called a *pushout*.

More succinctly we will set  $j_N := q \circ \iota_N$  and  $j_{N'} := q \circ \iota_{N'}$ . This gives a new cospan



Importantly, our pushout has a universal property which will come into play later on. Given any set  $Q$  and maps  $\tau : N \rightarrow Q$ ,  $\tau' : N' \rightarrow Q$  such that for all  $y \in Y$ ,  $\tau(o_N(y)) = \tau'(i_{N'}(y))$ , there exists a unique  $u$  such that the following diagram commutes.



## 1.2 Bicategories

The goal of our thesis is to lift the framework presented above to a 2-categorical level, allowing us to view Markov processes as morphisms, and establishing arrows between said morphisms. These so called 2-cells will be *simulations*, defined later. For now, we present some background on bicategories, most of which is taken from [11].

A bicategory is a particular notion of weak 2-category. In a bicategory, the hom-sets are in fact categories themselves. The associativity and unit laws of enriched categories, however, hold only up to isomorphism. As such it is weakly enriched over **Cat**. More specifically:

**Definition 1.** A bicategory  $\mathcal{B}$  consists of the following data:

- A collection  $ob(\mathcal{B})$ , with elements called 0-cells  $A, B, \dots$
- For each pair of elements  $A, B$ , a category  $\mathcal{B}(A, B)$  which in turn has objects 1-cells  $f, g, \dots$  and for morphism 2-cells  $\alpha, \beta, \dots$

- For each triple of 0-cells  $A, B, C$ , a functor

$$c_{A,B,C} : \mathcal{B}(B, C) \times \mathcal{B}(A, B) \rightarrow \mathcal{B}(A, C)$$

which on objects takes a pair  $(g, f)$  and returns their *horizontal* composition  $g \circ f$ , and on morphisms takes a pair  $(\alpha, \beta)$  and returns their *horizontal* composition  $\beta * \alpha$ .

- For every 0-cell, a functor  $I_A : 1 \rightarrow \mathcal{B}(A, A)$  which effectively chooses an identity in the category  $\mathcal{B}(A, A)$
- Natural Isomorphisms

$$\begin{array}{ccc} \mathcal{B}(C, D) \times \mathcal{B}(B, C) \times \mathcal{B}(A, B) & \xrightarrow{1 \times c_{ABC}} & \mathcal{B}(C, D) \times \mathcal{B}(A, C) \\ \downarrow c_{BCD} \times 1 & \nearrow a_{ABCD} & \downarrow c_{ACD} \\ \mathcal{B}(B, D) \times \mathcal{B}(A, B) & \xrightarrow{c_{ABD}} & \mathcal{B}(A, D) \end{array}$$

- Natural Isomorphisms:

$$\begin{array}{ccc} \mathcal{B}(A, B) \times 1 & & \\ \downarrow 1 \times I_A & \nearrow r_{AB} & \searrow \cong \\ \mathcal{B}(A, B) \times \mathcal{B}(A, A) & \xrightarrow{c_{AAB}} & \mathcal{B}(A, B) \end{array}$$

and

$$\begin{array}{ccc} 1 \times \mathcal{B}(A, B) & & \\ \downarrow I_B \times 1 & \nearrow l_{AB} & \searrow \cong \\ \mathcal{B}(B, B) \times \mathcal{B}(A, B) & \xrightarrow{c_{ABB}} & \mathcal{B}(A, B) \end{array}$$

such that the following diagrams commute:

- The pentagon identity:

$$\begin{array}{ccccc}
 & ((kh)g)f & \xrightarrow{a*1} & (k(hg))f & \\
 & \swarrow a & & \searrow a & \\
 (kh)(gf) & & & & k((hg)f) \\
 & \searrow a & & \swarrow 1*a & \\
 & k(h(gf)) & & & 
 \end{array}$$

- The triangle identity:

$$\begin{array}{ccc}
 (gI)f & \xrightarrow{a} & g(I f) \\
 \searrow 1*r & & \swarrow 1*l \\
 & gf & 
 \end{array}$$

Note that the notion of *bicategory* is weaker than the more natural notion of 2-category, where all of the natural isomorphisms  $a, l, r$  are identities.

The most obvious example of a bicategory is the category **Cat** itself (it is in fact a strict 2-category). In this case, the 0-cells are categories themselves, the 1-cells functors, and the 2-cells are natural transformations.

We will use bicategories to formalize the notion that labelled transition systems and Markov processes can be composed, but that this composition is only associative up to isomorphism. We will develop the notion of *simulation* and show how they can be seen as 2-cells in our bicategorical structures.

# Chapter 2

## Labelled Transition Systems

Our first example of the theory presented in Section 1 will be labelled transition systems.

**Definition 2.** Given a set of states  $S$ , and a set of labels  $\mathcal{L}$ , a *labelled transition system* on  $S$  is a tuple  $(S, \alpha_S)$ , where  $\alpha_S$  is a map  $\alpha_S : S \rightarrow \mathcal{P}(S)^{\mathcal{L}}$ . Currying, we can think of  $\alpha_S$  as taking in both a state  $s$ , and a label  $\ell$ , and returning the set of states reachable from  $s$  by action  $\ell$ .

As in chapter 1, we can view our labelled transition systems as morphisms between input and output sets, which one should think of as nodes to respectively “enter” and “exit” the system.

**Definition 3.** Given two sets  $X, Y$ , a *labelled transition system* (LTS) from  $X$  to  $Y$  is a labelled transition system  $(S, \alpha_S)$  as above, as well as two injective morphisms  $i : X \rightarrow S$  and  $o : Y \rightarrow S$  called *input* and *output*.

We should think of an outside observer, allowed to influence the system  $S$  using actions labelled by  $\mathcal{L}$  which result in a non-deterministic (as opposed to probabilistic) response by

the system; the set of possible responses to performing the action labelled by  $\ell$  at state  $s$  is given by  $\alpha_S(s, \ell)$ .

Henceforth, we will be considering the case where  $|\mathcal{L}| = 1$  for simplicity, as it does not affect the theory but allows for cleaner exposition, and therefore write  $\alpha_S(s)$  for  $\alpha_S(s, \ell)$

## 2.1 Viewing LTSs as morphisms

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

Let us first give an intuition for this composition: this corresponds to cascading the transition systems one after the other by identifying states that were outputs in the first system with inputs in the second system, using  $Y$  to mediate this identification. Consider three sets  $X, Y, Z$  and two LTSs

$$(S, \alpha_S) : X \rightarrow Y$$

$$(T, \alpha_T) : Y \rightarrow Z$$

The category *Set* which has sets as object and functions as morphisms admits pushouts. Let us denote  $S +_Y T$  the pushout of  $S$  and  $T$  along  $i_T$  and  $o_S$ , and let  $j_S$  and  $j_T$  be the inclusion maps (as in Section 1).



$$\begin{array}{ccc}
Y & \xrightarrow{o_S} & S \\
i_T \downarrow & & \downarrow j_S \\
T & \xrightarrow{j_T} & S +_Y T
\end{array}$$

The composition of  $(S, \alpha_S)$  from  $X$  to  $Y$ , with  $(T, \alpha_T)$  from  $Y$  to  $Z$  is the LTS with input  $X$  and output  $Z$  defined as follows.

$$T * S := (S +_Y T, \beta)$$

where, for  $z \in S +_Y T$

$$\beta(z) = \mathcal{P}(j_S) \circ \alpha_S \circ j_S^{-1}(z) \cup \mathcal{P}(j_T) \circ \alpha_T \circ j_T^{-1}(z)$$

where in the above we allow a slight abuse of notation and conflate  $j_S^{-1}(z)$  with the unique element in this set if it exists. If either  $j_S^{-1}(z)$  is empty, or  $j_T^{-1}(z)$  (note that both cannot happen simultaneously), we simply ignore this term in the above definition.

Intuitively, the “dynamics” of  $(S +_Y T, \beta)$  at a state  $z$  is obtained by “combining” the dynamics of both  $S$  and  $T$  coherently.

The following diagram gives an intuition for our above definition. Note that it does not commute, but simply gives the domain and codomain of the maps used above.

$$\begin{array}{ccccc}
& & S & \xrightarrow{\alpha_S} & \mathcal{P}(S) \\
& \nearrow & \searrow j_S & & \searrow \mathcal{P}(j_S) \\
Y & & & & \\
& \searrow & \nearrow j_T & & \nearrow \mathcal{P}(j_T) \\
& & T & \xrightarrow{\alpha_T} & \mathcal{P}(T) \\
& & \nearrow & \xrightarrow{\beta} & \mathcal{P}(S +_Y T)
\end{array}$$

Again, we stress that composition here is associative only up to isomorphism. Given any state set  $X$ , the “identity”  $1_X$  will be  $(X, \alpha_X)$ , where for all  $x \in X$ ,  $\alpha_X(x) = \emptyset$ . Intuitively this is clear. Cascading a morphism  $(S, \alpha_S) : X \rightarrow Y$  with the morphism  $1_Y : Y \rightarrow Y$  in which no transitions occur should return a system isomorphic to  $(S, \alpha_S)$ .

## 2.2 Simulations as Morphisms between LTSs

Given two LTSs 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 the *bicategorical* level.

**Definition 4.** Given two LTSs  $(S, \alpha_S)$  and  $(T, \alpha_T)$  defined with the same input and output sets, a *simulation* of  $S$  by  $T$  is a function  $f : S \rightarrow T$  on the state spaces such that

$$f \circ i_S = i_T \text{ and } f \circ o_S = o_T$$

and the following diagram commutes in a lax way:

$$\begin{array}{ccc} S & \xrightarrow{\alpha_S} & \mathcal{P}(S) \\ \downarrow f & \wr & \downarrow \mathcal{P}(f) \\ T & \xrightarrow{\alpha_T} & \mathcal{P}(T) \end{array}$$

In such a case we say that  $(T, \alpha_T)$  simulates  $(S, \alpha_S)$ , or more simply that  $T$  simulates  $S$ .

Note that the usual definition of simulation uses relations rather than functions as we have done, but is easier for our purposes to use the above definition. Indeed, using this definition

makes it easier to construct a bicategory. It is not clear if the relational definition would also allow for a bicategorical treatment.

Given two finite sets  $X$  and  $Y$ , we'd like to have the “hom-set”  $\mathbf{LTS}(X, Y)$  of the previously defined “category”  $\mathbf{LTS}$ . But as indicated previously, this does not work out.

However, it is possible to construct a *bicategory*. We'll let the set  $\mathbf{LTS}(X, Y)$  be a category with objects the LTSs from  $X$  to  $Y$  and as morphisms simulations between such LTSs. We carry out this construction in the next section.

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.

The fact that the composition of two simulations  $f : S \rightarrow T$  and  $g : T \rightarrow P$  gives a new simulation is easily verified chasing the following diagram:

$$\begin{array}{ccc}
 S & \xrightarrow{\alpha_S} & \mathcal{P}(S) \\
 \downarrow f & \wr & \downarrow \mathcal{P}(f) \\
 T & \xrightarrow{\alpha_T} & \mathcal{P}(T) \\
 \downarrow g & \wr & \downarrow \mathcal{P}(g) \\
 P & \xrightarrow{\alpha_P} & \mathcal{P}(P)
 \end{array}$$

Given a LTS  $(S, \alpha_S)$ , the map  $\text{id}_S$  is indeed a simulation, and will be an identity for our simulation composition.

As such, given two sets  $X, Y$ , we can denote by  $\mathbf{LTS}(X, Y)$  the *category* of all labelled transition systems from  $X$  to  $Y$ .

We can now provide some intuition for the earlier construction of  $T * S$ . Indeed we set up  $\beta$  so that it was the “minimal” map which made the following commute laxly:

$$\begin{array}{ccccc}
 & & S & \xrightarrow{\alpha_S} & \mathcal{P}(S) \\
 & \nearrow & \searrow j_S & \wr & \searrow \mathcal{P}(j_S) \\
 Y & & S +_Y T & \xrightarrow{\beta} & \mathcal{P}(S +_Y T) \\
 & \searrow & \nearrow j_T & \wr & \nearrow \mathcal{P}(j_T) \\
 & & T & \xrightarrow{\alpha_T} & \mathcal{P}(T)
 \end{array}$$

In other words,  $\beta$  is the minimal map which allows  $(S +_Y T, \beta)$  to simulate both  $(S, \alpha_S)$  and  $(T, \alpha_T)$ !

## 2.3 The Bicategory LTS

To reach the bicategorical level, for every triple of sets  $X, Y$  and  $Z$  we introduce a functor

$$c_{XYZ} : \mathbf{LTS}(Y, Z) \times \mathbf{LTS}(X, Y) \rightarrow \mathbf{LTS}(X, Z)$$

.

Given two LTSs  $(S, \alpha_S) : X \rightarrow Y$  and  $(T, \alpha_T) : Y \rightarrow Z$ ,  $c_{XYZ}(S, T)$  is their composition  $S * T$  defined earlier. This specifies our functor on objects.

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

$$(S_k, \alpha_{S_k}) : X \rightarrow Y$$

and

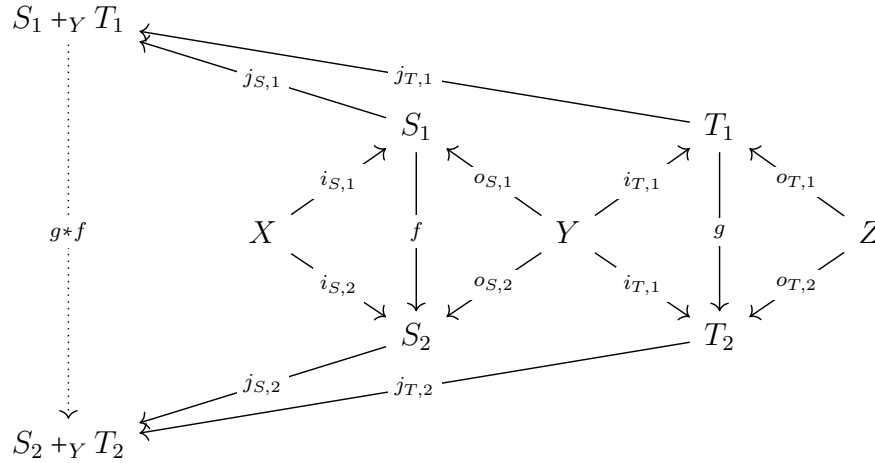
$$(T_k, \alpha_{T_k}) : Y \rightarrow Z$$

as well as two simulations

$$f : S_1 \Rightarrow S_2 \quad \text{and} \quad g : T_1 \Rightarrow T_2$$

Let us again denote  $j_{S,k} : S_k \rightarrow S_k +_Y T_k$  and  $j_{T,k} : T_k \rightarrow S_k +_Y T_k$  the pushout maps obtained by performing the horizontal composition  $S_k * T_k$ .

The horizontal composition  $c_{XYZ}(g, f) : T_1 * S_1 \Rightarrow T_2 * S_2$  is denoted  $g * f$  and is defined as the unique map making the following diagram commute:



Note that for  $z \in S_1 +_Y T_1$ ,

$$(g * f)(z) = \begin{cases} j_{T,2} \circ g(t) & \text{if } \exists t \in T_1 \text{ such that } z = j_{T,1}(t) \\ j_{S,2} \circ f(s) & \text{if } \exists s \in S_1 \text{ such that } z = j_{S,1}(s) \end{cases}$$

This horizontal composition is well defined. Let  $z \in S_1 +_Y T_1$ . Assume that there exists  $t \in T_1$  such that  $z = j_{T,1}(t)$  and  $s \in S_1$  such that  $z = j_{S,1}(s)$ . By definition of the pushout,

there exists  $y \in Y$  such that  $s = o_{S,1}(y)$  and  $t = i_{T,1}(y)$ . Then:

$$\begin{aligned}
(j_{S,2} \circ f)(s) &= (j_{S,2} \circ f \circ o_{S,1})(y) \\
&= (j_{S,2} \circ o_{S,2})(y) \quad \text{as } f \text{ is a simulation} \\
&= (j_{T,2} \circ i_{T,2})(y) \quad \text{using the pushout} \\
&= (j_{T,2} \circ g \circ i_{T,1})(y) \quad \text{as } g \text{ is a simulation} \\
&= (j_{T,2} \circ g)(t)
\end{aligned}$$

**Lemma 5.** The horizontal composition  $g * f$  is a simulation.

*Proof.* In order to prove that it is indeed a simulation, we have first to prove that  $(g * f) \circ j_{S,1} \circ i_{S,1} = j_{S,2} \circ i_{S,2}$ , and similarly for the output maps. But this is obvious when we consider the above diagram.

The second condition to verify is that the following diagram laxly commutes:

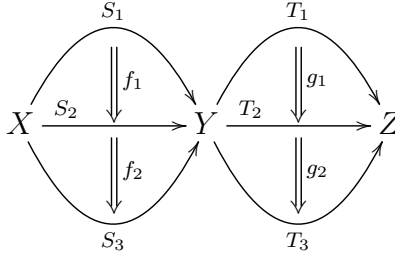
$$\begin{array}{ccc}
(S_1 +_Y T_1, \beta_1) & \xrightarrow{\beta_1} & \mathcal{P}(S_1 +_Y T_1) \\
\downarrow (g*f) & \supseteq & \downarrow \mathcal{P}(g*f) \\
(S_2 +_Y T_2, \beta_2) & \xrightarrow{\beta_2} & \mathcal{P}(S_2 +_Y T_2)
\end{array}$$

We'll examine the case where there exists  $t \in T_1$  such that  $z = j_{T,1}(t)$  and there exists  $s \in S_1$  such that  $z = j_{S,1}(s)$ . If either of  $j_{T,1}^{-1}(\{z\})$  or  $j_{S,1}^{-1}(\{z\})$  is empty, the computation is simpler.

$$\begin{aligned}
\beta_2((g * f)(z)) &= [\mathcal{P}(j_{S,2}) \circ \alpha_{S,2} \circ j_{S,2}^{-1}]((g * f)(z)) \cup [\mathcal{P}(j_{T,2}) \circ \alpha_{T,2} \circ j_{T,2}^{-1}]((g * f)(z)) \\
&= [\mathcal{P}(j_{S,2}) \circ \alpha_{S,2} \circ f](s) \cup [\mathcal{P}(j_{T,2}) \circ \alpha_{T,2} \circ g](t) \\
&\supseteq [\mathcal{P}(j_{S,2}) \circ \mathcal{P}(f) \circ \alpha_{S,1}](s) \cup [\mathcal{P}(j_{S,2}) \circ \mathcal{P}(g) \circ \alpha_{T,1}](t) \\
&= \mathcal{P}(g * f)(\mathcal{P}(i_{S,1}) \circ \alpha_{S,1} \circ i_{S,1}^{-1}(z)) \cup \mathcal{P}(g * f)(\mathcal{P}(i_{T,1}) \circ \alpha_{T,1} \circ i_{T,1}^{-1}(z)) \\
&= \mathcal{P}(g * f)(\mathcal{P}(i_{S,1}) \circ \alpha_{S,1} \circ i_{S,1}^{-1}(z) \cup \mathcal{P}(i_{T,1}) \circ \alpha_{T,1} \circ i_{T,1}^{-1}(z)) \\
&= \mathcal{P}(g * f)(\beta_1(z))
\end{aligned}$$

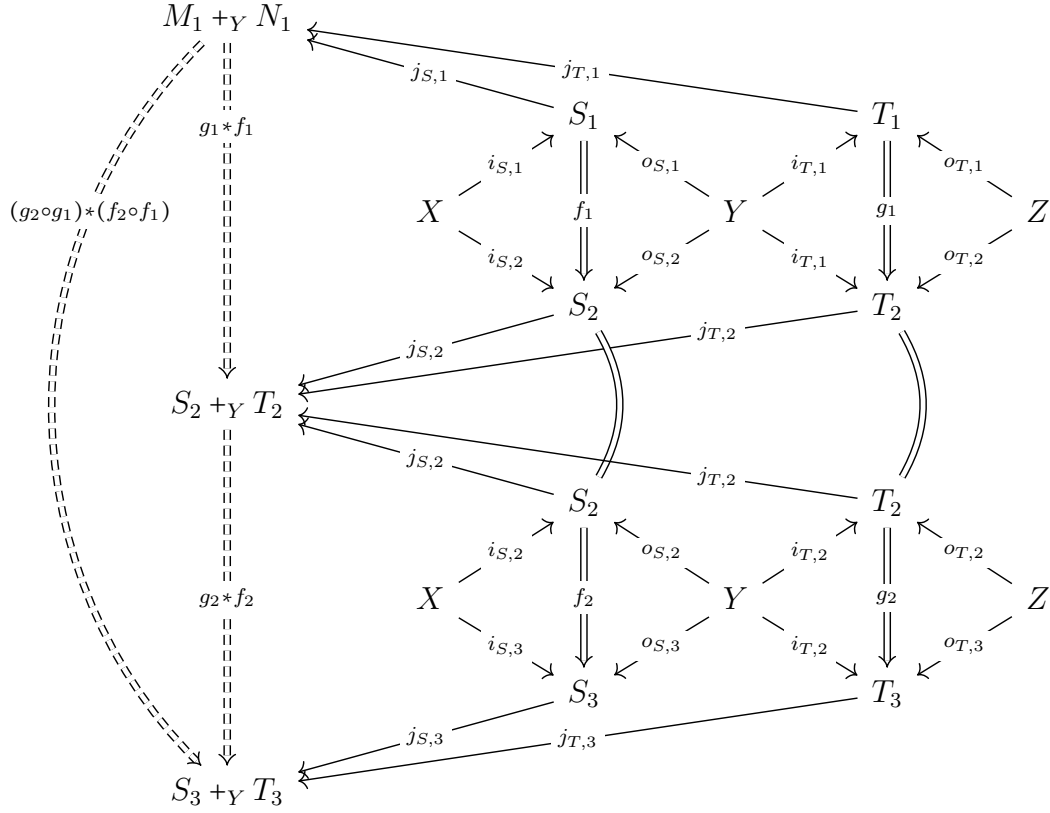
□

**Lemma 6.** The exchange law holds. That is, if we have the following picture



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

*Proof.* Again this can be proved purely diagrammatically. Note that  $g_1 * f_1$  is the unique map that makes the following “upper half” of the diagram commute, and that  $g_2 * f_2$  is the unique map that makes the “bottom half” commute. Now note that  $g_2 \circ g_1$  (resp.  $f_2 \circ f_1$ ) gives a simulation from  $T_1$  to  $T_3$  (resp. from  $S_1$  to  $S_3$ ), and we can chase the diagram to get the exchange law.



□

We must also show the existence of associators and unitors, satisfying the pentagon and triangle identities respectively. This will be pushed back to the section on Markov Processes.

We can now state the main result of this section.

**Theorem 7.** **LTS** is a bicategory.



# Chapter 3

## Markov Processes

We now turn our attention to a similar categorical treatment of Markov Processes, showing how viewing them as morphisms between input and output sets allows us to create a bicategory. We will be dealing with both discrete and continuous state spaces. Since the latter has some measure-theoretic details, we first work through the discrete case. It is pleasing that the measure theory and the category theory can be more or less “factored” into separate sections.

### 3.1 Discrete Markov Processes

**Definition 8.** Given a finite set  $M$ , a *Markov kernel* on  $M$  is a map  $\tau : M \times M \rightarrow [0, 1]$  such that for all  $m \in M$ ,  $\tau(m, \cdot)$  is a subprobability measure on  $M$ . A *labelled Markov process* on  $M$  is a collection  $(\tau_a)$  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 thesis. 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 [2] and in Section 2 we can view our labelled Markov processes as morphisms between input and output sets.

Again we denote **Set** the category with sets as objects and functions as morphisms.

**Definition 9.** 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  $i : X \rightarrow M$  and  $o : Y \rightarrow M$  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  $\tau_a$ , 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.

### 3.1.1 Viewing DLMPs as morphisms

As with labelled transition systems, we can compose Markov processes: again this corresponds to cascading the systems one after the other by identifying states that were outputs in the first DLMP with inputs in the second DLMP. Consider three sets  $X, Y, Z$  and two DLMPs

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

and

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

The pushout  $M +_Y N$  is constructed as in Section 2

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  $\tau^N$ .

Again we face a problem of associativity of composition up to isomorphism only. 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  $x, y \in X$ ,  $\tau_a(x, y) = 0$ .

### 3.1.2 Simulations between DLMPs

Given two Markov processes with the same input and output sets, it is again natural to ask whether they are related in some way or not. Let us introduce the notion of simulation between Markov Processes. This definition is slightly more cumbersome than in Section 2, as we are dealing with probability distributions over the state spaces.

**Definition 10.** 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  $f : N \rightarrow M$  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 we write  $f^{-1}(m)$  for  $f^{-1}(\{m\})$ . In such a case, we say that  $\mathcal{M}$  *simulates*  $\mathcal{N}$  and write  $f : \mathcal{N} \Rightarrow \mathcal{M}$ .

The composition of two simulations with the same input and output sets is again given by standard function composition denoted  $\circ$ . It remains 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  $g : \mathcal{M}_2 \Rightarrow \mathcal{M}_3$  with  $\mathcal{M}_k = (M_k, (\tau_a^k)_{a \in Act}, i_k, o_k) : X \rightarrow Y$ . 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.  $\square$

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

### 3.1.3 The Bicategory DLMP

Given a triple of sets  $X, Y$  and  $Z$ , we need a functor

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

representing horizontal composition.

Given two DLMPs  $\mathcal{M} : X \rightarrow Y$  and  $\mathcal{N} : Y \rightarrow Z$ ,  $c_{XYZ}(\mathcal{N}, \mathcal{M})$  is their composition  $\mathcal{N} * \mathcal{M}$  defined in Section 3.1.1.

Let us consider four DLMPs (with  $k = 1, 2$ ) :

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

and

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

as well as two simulations

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

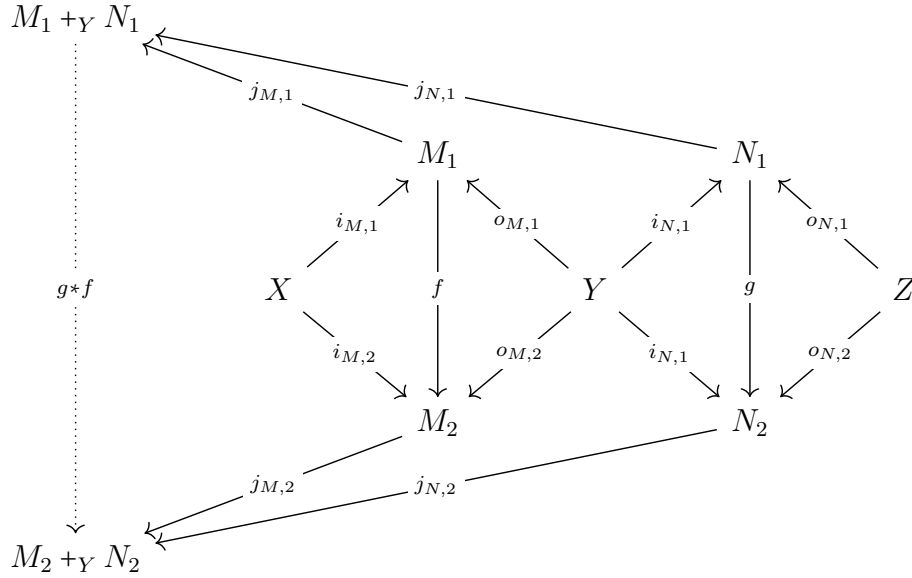
The horizontal composition  $(g * f)$  is obtained as in the previous chapter. This is interesting, as it suggests that there is a sort of “factorization” at play here; the dynamics of the systems (probabilistic vs. nondeterministic) does not affect how we obtain simulations when we compose them.

Indeed the horizontal composition  $(g * f) : \mathcal{N}_1 * \mathcal{M}_1 \Rightarrow \mathcal{N}_2 * \mathcal{M}_2$  is defined as follows. For  $m \in M_1 +_Y N_1$ ,

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

Again,  $g * f(m)$  is well defined.

Diagrammatically, the situation is the following :



**Lemma 11.** The horizontal composition  $g * f$  is a simulation.

*Proof.* 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  $m_2 \in M_2 +_Y N_2$ ,

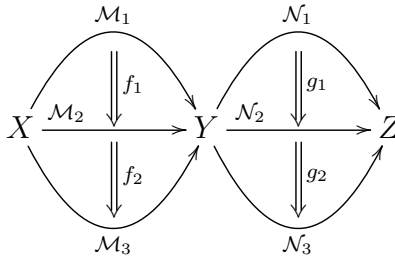
$$\tau_a^2((g * f)(m_1), m_2) \geq \tau_a^1(m_1, (g * f)^{-1}(m_2))$$

There are a few cases to check, but the proof is straightforward and as such we examine one case (the other cases are nearly identical). Suppose  $m_1 \in j_{M,1}(M_1)$  and that  $m_2 \in j_{M,2}(M_2)$ . Then:

$$\begin{aligned}\tau_a^1(m_1, (g \circ f)^{-1}(m_2)) &= \tau_a^1(m_1, (j_{M,1} \circ f^{-1} \circ j_{M,2}^{-1})(m_2)) \\ &= \tau_a^{M,1}(j_{M,1}^{-1}(m_1), (f^{-1} \circ j_{M,2}^{-1})(m_2)) \\ &\leq \tau_a^{M,2}(f \circ j_{M,1}^{-1}(m_1), j_{M,2}^{-1}(m_2)) \\ &= \tau_a^2((g \circ f)(m_1), m_2)\end{aligned}$$

☐

**Lemma 12.** The exchange law holds. Namely, when we have the following situation





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

*Proof.* This is exactly as in the previous case Lemma 6 □

**Lemma 13.** 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

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

*Proof.* Let us consider three DLMPs

$$\mathcal{M} = (M, (\tau_a^M), i_M, o_M) : W \rightarrow X$$

$$\mathcal{N} = (N, (\tau_a^N), i_N, o_N) : X \rightarrow Y$$

$$\mathcal{P} = (P, (\tau_a^P), i_P, o_P) : Y \rightarrow Z$$

We will construct the associator  $\alpha_{\mathcal{M}\mathcal{N}\mathcal{P}} : \mathcal{P} * (\mathcal{N} * \mathcal{M}) \Rightarrow (\mathcal{P} * \mathcal{N}) * \mathcal{M}$ , i.e. a simulation map

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

We will denote the pushout maps  $j_M^{M+YN} : M \rightarrow M +_Y N$  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:

$$\begin{array}{ccccc}
 X & \xrightarrow{i_N} & N & \xrightarrow{j_N^{N+Y P}} & N +_Y P \\
 \downarrow o_M & & \downarrow j_N^{M+X N} & & \downarrow j_{N+Y P}^{M+X(N+Y P)} \\
 M & \xrightarrow{j_M^{M+X N}} & M +_X N & \xrightarrow{\alpha_1} & M +_X (N +_Y P) \\
 & \searrow j_M^{M+X(N+Y P)} & & & \\
 & & & & M +_X (N +_Y P)
 \end{array}$$

By the universal property of the pushout  $M +_X (N +_Y P)$ , there exists a unique map

$$\alpha_1 : M +_X N \rightarrow M +_X (N +_Y P)$$

making the above diagram commute.

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

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

$$\begin{array}{ccccc}
 Y & \xrightarrow{o_N} & N & \xrightarrow{j_N^{M+X N}} & M +_X N \\
 \downarrow i_P & & & \searrow j_{M+X N}^{(M+X N)+Y P} & \downarrow \\
 P & \xrightarrow{j_P^{(M+X N)+Y P}} & (M +_X N) +_Y P & \xrightarrow{\alpha_1} & M +_X (N +_Y P) \\
 & \searrow j_P^{N+Y P} & & & \\
 & & N +_Y P & \xrightarrow{j_{N+Y P}^{M+X(N+Y P)}} & M +_X (N +_Y P)
 \end{array}$$

By the universal property of the pushout  $(M +_X N) +_Y P$ , there exists a unique map

$$(M +_X N) +_Y P \rightarrow M +_X (N +_Y P)$$

making this diagram commute. We call this map  $\alpha_{\mathcal{M}\mathcal{N}\mathcal{P}}$ . Note that we could have constructed the associator from the explicit definition of the pushout given in Section 3.1.1.

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.  $\square$

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  $\mathcal{M} : X \rightarrow Y$  a DLMP, we have

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

Pentagon identities and triangle identities are proven using similar computations. Hence:

**Theorem 14.** **DLMP** is a bicategory.

## 3.2 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.

### 3.2.1 LMP and simulation in the continuous case

**Definition 15.** Given a measurable space  $(M, \Sigma)$  a *Markov kernel* is a function  $\tau : M \times \Sigma \rightarrow [0, 1]$  where for each  $m \in M$  the function  $\tau(m, \cdot)$  is a subprobability measure on  $(M, \Sigma)$  and for each measurable set  $B \in \Sigma$  the function  $\tau(\cdot, B) : M \rightarrow [0, 1]$  is measurable where  $[0, 1]$  is equipped with the standard Borel-algebra. A *labelled Markov process* is a collection  $(\tau_a)$  of Markov kernels on  $(M, \Sigma)$  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 16.** 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  $(M, \Sigma)$  a measurable space, a labelled Markov Process  $(\tau_a)_{a \in Act}$ , an injective function  $i : X \rightarrow M$  and a function  $o : Y \rightarrow \Sigma$  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 17.** 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  $f : N \rightarrow M$  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  $a \in \text{Act}$ ,  $n \in N$  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  $f : \mathcal{N} \Rightarrow \mathcal{M}$ .

### 3.2.2 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 sets  $X, Y$ , there is a category **CLMP**( $X, Y$ ) which has as objects the CLMPs  $X \rightarrow Y$  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  $\mathcal{M} = (M, \Sigma, i_M, o_M, \tau^M) : X \rightarrow Y$  and  $\mathcal{N} = (N, \Lambda, i_N, o_N, \tau^N) : Y \rightarrow Z$ , there are two inclusion maps  $j_N : N \rightarrow M + N$  and  $j_M : M \rightarrow M + N$ . We then define the relation  $\sim$  on  $M + N$  as the smallest equivalence such that

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

We then define the quotient map  $q$  between measurable spaces

$$q : (M + N, \Sigma + \Lambda) \rightarrow ((M + N)/\sim, (\Sigma + \Lambda)/\sim)$$

where  $(\Sigma + \Lambda)/\sim$  is the smallest  $\sigma$ -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

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

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  $\tau'_a$  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  $\Sigma$  be the discrete  $\sigma$ -algebra on  $X$ , then the identity is

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

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  $f : \mathcal{M}_1 \Rightarrow \mathcal{M}_2 : X \rightarrow Y$  and  $g : \mathcal{N}_1 \Rightarrow \mathcal{N}_2 : Y \rightarrow Z$  where  $\mathcal{M}_k =$

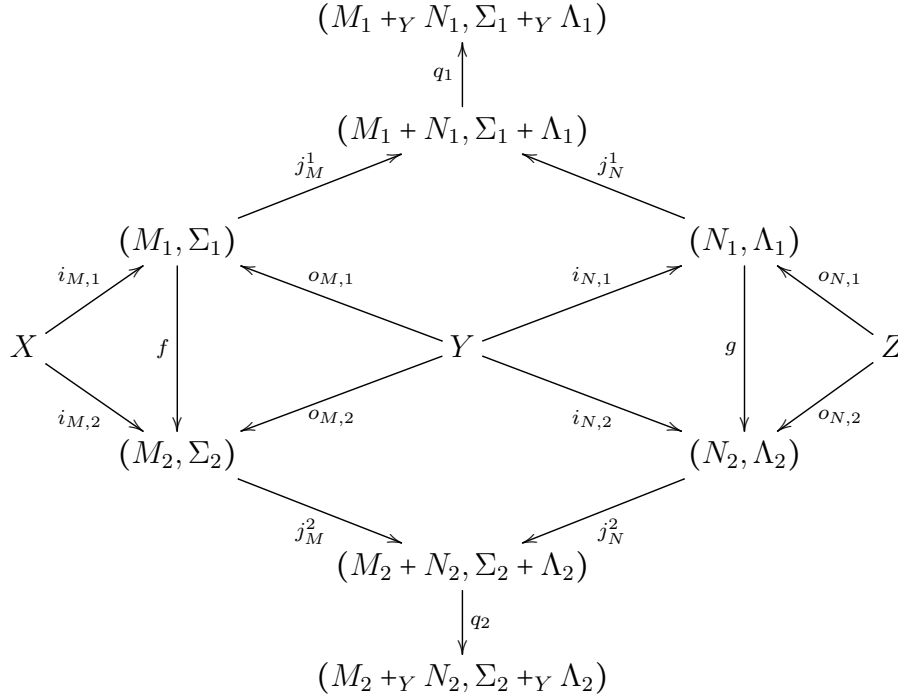
$(M_k, \Sigma_k, \tau^{M,k}, i_{M,k}, o_{M,k})$  and  $\mathcal{N}_k = (N_k, \Lambda_k, \tau^{N,k}, i_{N,k}, o_{N,k})$  ( $k = 1, 2$ ). We use similar notations as for the composition of CLMPs but index them by 1 or 2 (see following diagram).

We define their horizontal composition as

$$g * f : M_1 +_Y N_1 \rightarrow M_2 +_Y N_2$$

$$n \mapsto q_2 \circ j_N^2 \circ g(n') \text{ if } \exists n' \in N_1 \ n = q_1 \circ j_N^1(n')$$

$$m \mapsto q_2 \circ j_M^2 \circ f(m') \text{ if } \exists m' \in M_1 \ m = q_1 \circ j_M^1(m')$$



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. This gives the main result of the chapter:

**Theorem 18.** **CLMP** is a bicategory.



# Chapter 4

## Conclusion

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 thesis 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. [12] 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 the one presented in this thesis, but clearly it is an interesting topic to be examined.

# Bibliography

- [1] F. Clerc, H. Humphrey, and P. Panangaden, “Bicategories of Markov processes.” Kim Larsen Festschrift, 2017.
- [2] J. C. Baez, B. Fong, and B. S. Pollard, “A compositional framework for Markov processes.” ArXiv:1508.06448, Sept 2015.
- [3] J. R. Norris, *Markov Chains*. Cambridge Series in Statistical and Probabilistic Mathematics, Cambridge University Press, 1997.
- [4] B. Fong, “Decorated cospans,” *Theory and Applications of Categories*, vol. 30, pp. 1096–1120, 2015.
- [5] R. Milner, *A Calculus for Communicating Systems*, vol. 92 of *Lecture Notes in Computer Science*. Springer-Verlag, 1980.
- [6] R. Milner, *Communication and Concurrency*. Prentice-Hall, 1989.
- [7] D. Park, “Concurrency and automata on infinite sequences,” in *Proceedings of the 5th GI Conference on Theoretical Computer Science*, no. 104 in *Lecture Notes In Computer Science*, pp. 167–183, Springer-Verlag, 1981.
- [8] K. G. Larsen and A. Skou, “Bisimulation through probablistic testing,” *Information and Computation*, vol. 94, pp. 1–28, 1991.

- [9] P. Panangaden, *Labelled Markov Processes*. Imperial College Press, 2009.
- [10] J. Desharnais, A. Edalat, and P. Panangaden, “Bisimulation for labeled Markov processes,” *Information and Computation*, vol. 179, pp. 163–193, Dec 2002.
- [11] T. Leinster, “Basic bicategories,” 1998.
- [12] M. Mislove, J. Ouaknine, D. Pavlovic, and J. Worrell, “Duality for labelled Markov processes,” in *Foundations of Software Science and Computation Structures, FOSSACS* (I. Walukiewicz, ed.), vol. 2987 of *Lecture Notes In Computer Science*, pp. 393–407, 2004.