The compositional construction of Markov processes II

L. de Francesco Albasini N. Sabadini R.F.C. Walters

October 30, 2018

1 Introduction

In [4] we introduced a notion of Markov automaton, together with parallel operations which permit the compositional description of Markov processes. We illustrated by showing how to describe a system of n dining philosophers (with 12^n states), and we observed that Perron-Frobenius theory yields a proof that the probability of reaching deadlock tends to one as the number of steps goes to infinity. In this paper we add sequential operations to the algebra (and the necessary structure to support them) following analogous developments in [9, 5]. The extra operations permit the description of hierarchical systems, and ones with evolving geometry. We illustrate our algebra by describing a system called Sofia's Birthday Party, originally introduced in [9].

There is a huge literature on probabilistic and weighted automata, transducers, and process calculi (see for example, [2, 7, 10, 12, 14, 6]). However the model of [4] is distinguished from the others in the following ways:

(i) In other probabilistic automata models ([12],[1]) the sum of probabilities of actions out of a given state *with a given label* is 1. This means that the probabilities are conditional on the existence of the label. In our model the sum of probabilities of actions out of a given state *for all labels* is 1. The reason for this is further explained by the next point.

(ii) Our weighted automata are very close mathematically to weighted transducers but conceptually very different. Instead of modelling devices which translate input to output, the idea is we model devices with number of parallel interfaces, and when a transition occurs in the device this induces a transition on each of the interfaces (the interfaces are part of the device). In order to have binary operations of composition the interfaces are divided into left and right interfaces. The notions of initial and final states are also generalized in our notion of weighted automaton to become instead functions into the state space. These sequential interfaces are not to be thought of as initial and final states, but hooks into the state space at which a behaviour may enter or leave the device. The application of our weighted automata is to concurrent hierarchical and distributed systems rather than language recognition or processing. In [5] we show how data types and also state on the connectors (shared variables) may be added to our model.

(iii) For many compositional models the communication is based on process algebras like CCS, CSP. Instead ours models truly concurrent systems with explicit network topologies. One of the key aspects is our algebra of connectors: parallel and sequential "wires", which give an hierarchical network topology to expressions in the algebra.

(iv) Our automata both with respect to the sequential and with respect to the parallel operations form the arrows of a symmetric monoidal categories, with other well-known categorical properties (see, for example, [3]). The operations are in fact based on the operations available in the categories of spans and cospans. The categorical operations have an associated geometry. Important equations satisfied are the Frobenius and separable axioms [13].

As we have said the novelty of this paper is the introduction of *sequential operations* to the algebra of [4]. This requires extra structure to be added to the automata, namely sequential interfaces.

Further, in [5] weighted automata (where the weighting of a transition is a non-negative real number) played a subsidiary role. However for sequential composition weighted automata are more fundamental, since in identifying states of two different automata it is the relative weight given to decisions which is important rather than the probabilities. Technically this appears in the fact that normalization is not compositional with respect to sequential operations, whereas for parallel operations it is. For a summary of recent work on the various kinds of weighted automata see [6], and in particular [11].

Another aim of the paper is to illustrate how hierarchical and mobile systems may be modelled in this algebra, using the combined sequential and parallel operations. Given a set of automata S, let us denote the set of automata given as expressions in terms of parallel operations in the automata S as $\Pi(S)$ and given as expressions in terms of sequential operations as $\Sigma(S)$. Let E be the set of elementary automata with only one transition. Then any automaton has a representation in $\Sigma(E)$. The dining philosopher problem of [5] is described as an element of $\Pi\Sigma(E)$, that is of communicating sequential systems. An element of $\Sigma\Pi\Sigma(E)$ is one in which the parallel structure may evolve, and so on. The system Sofia's Birthday Party is in $\Pi\Sigma\Sigma(E)$ but illustrates also the form of systems of type $\Pi\Sigma\Pi\Sigma(E)$. The power of the sequential and parallel operations is that they may be alternated, as the alternating quantifiers in logic, or the alternation in alternating Turing machines, or the alternating sums and products in State Charts.

2 Weighted automata with parallel and sequential interfaces

In this section we define weighted and Markov automata *with sequential and parallel interfaces*, which however we shall call just weighted and Markov automata. The reader should be aware that the definitions of [5] differ in lacking sequential interfaces. We also do not require here for weighted automata the special symbols ε and the condition that the rows of the total matrix are strictly positive: we reserve those conditions for what we now call *positive weighted automata*.

Notice that in order to conserve symbols in the following definitions we shall use the same symbol for the automaton, its state space and its family of matrices of transitions, distinguishing the separate parts only by the font.

Definition 2.1 *Consider two finite alphabets A and B, and two finite sets X and Y. A* weighted automaton $\mathbf{Q}$ with left parallel interface A, right parallel interface B, top sequential interface X, *and* bottom sequential interface Y, *consists of a finite set Q of states, and an $A \times B$ indexed family $\mathsf{Q} = (\mathsf{Q}_{a,b})_{(a\in A, b\in B)}$ of $Q \times Q$ matrices with non-negative real coefficients, and two functions, $\gamma_0 : X \to Q$, and $\gamma_1 : Y \to Q$. We denote the elements of the matrix $\mathsf{Q}_{a,b}$ by $[\mathsf{Q}_{a,b}]_{q,q'}$ $(q, q' \in Q)$.*

We call the matrix $\mathcal{Q} = \sum_{a\in A, b\in B} \mathsf{Q}_{a,b}$ the *total matrix* of the automaton.

We will use a brief notation for the automata $\mathbf{Q}$ indicating its interfaces, namely $\mathbf{Q}^X_{Y;A,B}$. We shall use the same symbols γ_0, γ_1 for the sequential interface functions of any automata, and we will sometimes refer to these functions as the sequential interfaces. Notice that the terms 'left', 'right', 'top' and 'bottom' for the interfaces have no particular semantic significance - they are chosen to be semantically neutral in order not to suggest input, output, initial or final.

Definition 2.2 *A weighted automaton $\mathbf{Q}$ is* positive *if the parallel interfaces A and B contain special elements, the symbols ε_A and ε_B, and satisfies the property that the row sums of the matrix $\mathsf{Q}_{\varepsilon_A,\varepsilon_B}$ are strictly positive.*

For a positive weighted automaton the total matrix has strictly positive row sums.

Definition 2.3 *A* Markov automaton $\mathbf{Q}$ with left interface A, right interface B, top sequential interface X, *and* bottom sequential interface Y, *written briefly $\mathbf{Q}^X_{Y;A,B}$, is a positive weighted automaton satisfying the extra condition that the row sums of the total matrix $\mathcal{Q}$ are all 1. That is, for all q*

$$\sum_{q'} \sum_{a\in A, b\in B} [\mathsf{Q}_{a,b}]_{q,q'} = 1.$$

For a Markov automaton we call $[\mathsf{Q}_{a,b}]_{q,q'}$ the *probability of the transition from q to q' with left signal a and right signal b.*

The idea is that in a given state various transitions to other states are possible and occur with various probabilities, the sum of these probabilities being 1. The transitions that occur have effects, which we may think of a *signals*, on the two interfaces of the automaton, which signals are represented by letters in the alphabets. We repeat that it is fundamental *not* to regard the letters in A and B as inputs or outputs, but rather signals induced by transitions of the automaton on the interfaces. For examples see section 2.

When both A and B are one element sets and $X = Y = \emptyset$ a Markov automaton is just a Markov matrix.

Definition 2.4 *Consider a weighted automaton $\mathbf{Q}$ with interfaces A and B. A* behaviour *of length k of $\mathbf{Q}$ consists of two words of length k, one $u = a_1 a_2 \cdots a_k$ in A^* and the other $v = b_1 b_2 \cdots b_k$ in B^* and a sequence of non-negative row vectors*

$$x_0,\ x_1 = x_0 \mathsf{Q}_{a_1,b_1},\ x_2 = x_1 \mathsf{Q}_{a_2,b_2},\ \cdots,\ x_k = x_{k-1} \mathsf{Q}_{a_k,b_k}.$$

Notice that, even for Markov automata, x_i is not *generally a distribution of states; for example often $x_i = 0$.*

Definition 2.5 *The* normalization *of a positive weighted automaton $\mathbf{Q}$, denoted $\mathbf{N}(\mathbf{Q})$ is the Markov automaton with the same interfaces and states, but with*

$$\left[\mathsf{N}(\mathsf{Q})_{a,b}\right]_{q,q'} = \frac{[\mathsf{Q}_{a,b}]_{q,q'}}{\sum_{q' \in Q} [\mathcal{Q}]_{q,q'}} = \frac{[\mathsf{Q}_{a,b}]_{q,q'}}{\sum_{q' \in Q} \sum_{a \in A, b \in B} [\mathsf{Q}_{a,b}]_{q,q'}}.$$

It is obvious that a weighted automaton $\mathbf{Q}$ is Markov if and only if $\mathbf{Q} = \mathbf{N}(\mathbf{Q})$.

Definition 2.6 *If $\mathbf{Q}$ is a weighted automaton and k is a natural number, then the* automaton of k step paths *in $\mathbf{Q}$, which we denote as $\mathbf{Q}^k$ is defined as follows: the states of $\mathbf{Q}^k$ are those of $\mathbf{Q}$; the sequential interfaces are the same; the left and right interfaces are A^k and B^k respectively. If $u = (a_1, a_2, \cdots, a_k) \in A^k$ and $v = (b_1, b_2, \cdots, b_k) \in B^k$ then*

$$(\mathsf{Q}^k)_{u,v} = \mathsf{Q}_{a_1,b_1} \mathsf{Q}_{a_2,b_2} \cdots \mathsf{Q}_{a_k,b_k}.$$

The definition for positive weighted automata requires in addition that $\varepsilon_{A^k} = (\varepsilon_A, \cdots, \varepsilon_A), \varepsilon_{B^k} = (\varepsilon_B, \cdots, \varepsilon_B)$.

If $\mathbf{Q}$ is a weighted automaton and $u = (a_1, a_2, \cdots, a_k) \in A^k$, $v = (b_1, b_2, \cdots, b_k) \in B^k$, then $[(\mathsf{Q}^k)_{u,v}]_{q,q'}$ is the sum over all paths from q to q' with left signal sequence u and right signal sequence v of the weights of paths, where the weight of a path is the product of the weights of the steps.

2.1 Graphical representation of weighted automata

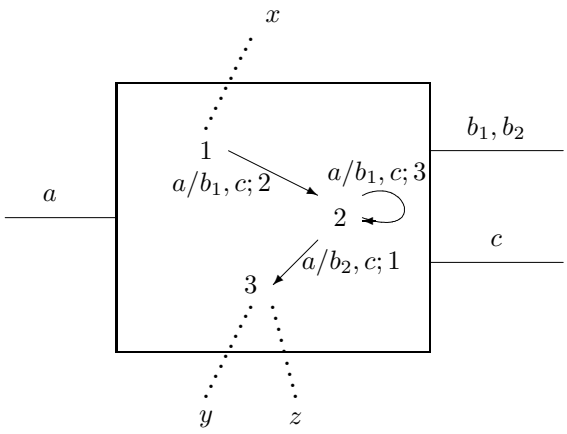
Although the definitions above are mathematically straightforward, in practice a graphical notation is more intuitive. We may compress the description of an automaton with parallel interfaces A and B, which requires $A \times B$ matrices, into a single labelled graph, like the ones introduced in [8]. We indicate by describing some examples.

2.1.1 An example

Consider the automaton with parallel interfaces $\{a\}$, $\{b_1, b_2\} \times \{c\}$, sequential interfaces $\{x\}$, $\{y, z\}$; with states $\{1, 2, 3\}$ sequential interface functions $x \mapsto 1$ and $y, z \mapsto 3$; and transition matrices

$$\mathsf{Q}_{a,(b_1,c)} = \begin{bmatrix} 0 & 2 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \ \mathsf{Q}_{a,(b_2,c)} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}.$$

This information may be put in the diagram:



The following two examples have both sequential interfaces $\emptyset$ and hence we will omit the sequential information.

2.1.2 A philosopher

Consider the alphabet $A = \{t, r, \varepsilon\}$. A philosopher is an automaton **Phil** with left interface A and right interfaces A, state space $\{1, 2, 3, 4\}$, both sequential interfaces $\emptyset \subseteq \{1, 2, 3, 4\}$, and transition matrices

$$\mathsf{Phil}_{\varepsilon,\varepsilon} = \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & \frac{1}{2} \end{bmatrix},$$

$$\mathsf{Phil}_{t,\varepsilon} = \begin{bmatrix} 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \ \mathsf{Phil}_{\varepsilon,t} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\mathsf{Phil}_{r,\varepsilon} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} \\ 0 & 0 & 0 & 0 \end{bmatrix}, \ \mathsf{Phil}_{\varepsilon,r} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 & 0 \end{bmatrix}.$$

The other four transition matrices are zero matrices.

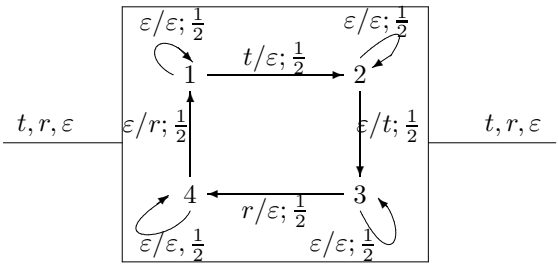
Notice that the total matrix of **Phil** is

$$\begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & 0 & 0 & \frac{1}{2} \end{bmatrix},$$

which is clearly stochastic, so **Phil** is a Markov automaton.

The intention behind these matrices is as follows: in all states the philosopher does a transition labelled ε, ε (*idle transition*) with probability $\frac{1}{2}$; in state 1 he does a transition to state 2 with probability $\frac{1}{2}$ labelled t, ε (*take the left fork*);

in state 2 he does a transition to state 3 with probability $\frac{1}{2}$ labelled ε, t (*take the right fork*); in state 3 he does a transition to state 4 with probability $\frac{1}{2}$ labelled r, ε (*release the left fork*); and in state 4 he does a transition to state 1 with probability $\frac{1}{2}$ labelled ε, r (*release the right fork*). All this information may be put in the following diagram.



2.1.3 A fork

Consider again the alphabet $A = \{t, r, \varepsilon\}$. A fork is an automaton **Fork** with left interface A and right interface A, state space $\{1, 2, 3\}$, both sequential interfaces $\emptyset \subseteq \{1, 2, 3\}$, and transition matrices

$$\mathsf{Fork}_{\varepsilon,\varepsilon} = \begin{bmatrix} \frac{1}{3} & 0 & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{2} \end{bmatrix},$$

$$\mathsf{Fork}_{t,\varepsilon} = \begin{bmatrix} 0 & \frac{1}{3} & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \ \mathsf{Fork}_{\varepsilon,t} = \begin{bmatrix} 0 & 0 & \frac{1}{3} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\mathsf{Fork}_{r,\varepsilon} = \begin{bmatrix} 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \ \mathsf{Fork}_{\varepsilon,r} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 \end{bmatrix}.$$

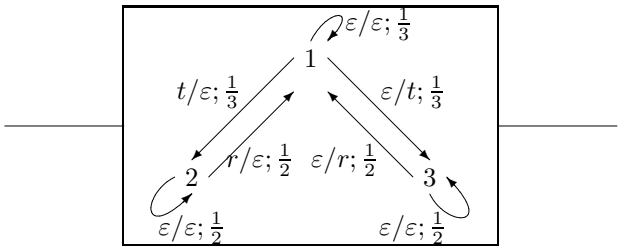
The other four transition matrices are zero.

Fork is a Markov automaton since its total matrix is

$$\begin{bmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} \end{bmatrix}.$$

The intention behind these matrices is as follows: in all states the fork does a transition labelled ε, ε (*idle transition*) with positive probability (either $\frac{1}{3}$ or $\frac{1}{2}$); in state 1 it does a transition to state 2 with probability $\frac{1}{3}$ labelled t, ε (*taken to the left*); in state 1 he does a transition to state 3 with probability $\frac{1}{3}$ labelled ε, t (*taken to the right*); in state 2 he does a transition to state 1 with probability $\frac{1}{2}$ labelled r, ε (*released to the left*); in state 3 he does a transition to state 1 with probability $\frac{1}{2}$ labelled ε, r (*released to the right*).

All this information may be put in the following diagram:



3 The algebra of weighted automata: operations

Now we define operations on weighted automata analogous (in a precise sense)to those defined in [8, 9].

3.1 Sequential operations

3.1.1 Sum

Definition 3.1 *Given weighted automata* $\mathbf{Q}^X_{Y;A,B}$ *and* $\mathbf{R}^Z_{W;C,D}$ *the* sum $\mathbf{Q} \boxplus \mathbf{R}$ *is the weighted automaton which has set of states the disjoint union* $Q + R$*, left interfaces* $A + C$*, right interface* $B + D$*, top interface* $X + Z$*, bottom interface* $Y + W$*, (all disjoint sums),* $\gamma_0 = \gamma_{0,\mathbf{Q}} + \gamma_{0,\mathbf{R}}$*,* $\gamma_1 = \gamma_{1,\mathbf{Q}} + \gamma_{1,\mathbf{R}}$*. The transition matrices are*

$$[(\mathsf{Q} + \mathsf{R})_{a,b}]_{q,q'} = [\mathsf{Q}_{a,b}]_{q,q'},$$
$$[(\mathsf{Q} + \mathsf{R})_{c,d}]_{r,r'} = [\mathsf{R}_{c,d}]_{r,r'},$$

all other values being 0*.*

3.1.2 Sequential composition

Definition 3.2 *Given weighted automata* $\mathbf{Q}^X_{Y;A,B}$ *and* $\mathbf{R}^Y_{Z;C,D}$*, the* sequential composite of weighted automata $\mathbf{Q} \circ \mathbf{R}$ *has set of states the equivalence classes of* $Q+R$ *under the equivalence relation generated by the relation* $\gamma_{1,\mathbf{Q}}(y) \sim \gamma_{0,\mathbf{R}}(y)$*,* $(y \in Y)$*. The left interface is the disjoint sum* $A + C$*, right interface* $B + D$*, the top interface is* X*, the bottom interface is* Z*. The interface functions are*

$$\gamma_0 = X \xrightarrow{\gamma_0} Q \to Q + R \to (Q + R)/\sim\,,\ \gamma_1 = Z \xrightarrow{\gamma_1} R \to Q + R \to (Q + R)/\sim\,.$$

Denoting the equivalence class of a state s *by* $[s]$ *the transition matrices are:*

$$[(\mathsf{Q} \circ \mathsf{R})_{a,b}]_{[q],[q']} = \sum_{s \in [q], s' \in [q']} [\mathsf{Q}_{a,b}]_{s,s'},$$
$$[(\mathsf{Q} \circ \mathsf{R})_{c,d}]_{[r],[r']} = \sum_{s \in [r], s' \in [r']} [\mathsf{R}_{c,d}]_{s,s'},$$

all other values being 0*.*

3.1.3 Sequential constants

Definition 3.3 *Given a relation $\rho \subset X + Y$ we define a weighted automaton* $\mathbf{Seq}(\rho)$ *as follows: it has state space the equivalence classes of $X + Y$ under the equivalence relation $\sim$ generated by ρ. It has parallel interfaces $\emptyset$, so there are no transition matrices. The sequential interfaces are $\gamma_0 : X \to (X + Y)/\sim$, $\gamma_1 : Y \to (X + Y)/\sim$, both functions taking an element to its equivalence class.*

Sequential connectors Some special cases have particular importance and are called *sequential connectors* or *wires*: (i) the automaton corresponding to the identity function 1_A, considered as a relation on $A \times A$ is also called 1_A; (ii) the automaton corresponding to the codiagonal function $\nabla : A+A \to A$ (considered as a relation) is called ∇; the automaton corresponding to the opposite relation ∇ is called ∇^o; (iii) the automaton corresponding to the function $twist : A\times B \to B\times A$ is called $twist_{A,B}$; (iv) the automaton corresponding to the function $\emptyset \subseteq A$ is called i; the automaton corresponding to the opposite relation of the function $\emptyset \subseteq A$ is called i^o. Notice that we have overworked the symbol ∇ and it will be used again in another sense; however the context should make clear which use we have in mind.

The role of sequential wires is to *equate states*.

The distributive law The bijection $\delta : X \times Y + X \times Z \to X \times (Y + Z)$ and its inverse $\delta^{-1} : X \times (Y + Z) \to X \times Y + X \times Z)$ considered as relations yield weighted automata which we will refer to with the same names.

3.2 Parallel operations

3.2.1 Parallel composition

Definition 3.4 *Given weighted automata $\mathbf{Q}^X_{Y;A,B}$ and $\mathbf{R}^Z_{W;C,D}$ the* parallel composite $\mathbf{Q} \times \mathbf{R}$ *is the weighted automaton which has set of states $Q \times R$, left interfaces $A \times C$, right interface $B \times D$, top interface $X \times Z$, bottom interface $Y \times W$, sequential interface functions $\gamma_{0,\mathbf{Q}} \times \gamma_{0,\mathbf{R}}$, $\gamma_{1,\mathbf{Q}} \times \gamma_{1,\mathbf{R}}$ and transition matrices,*

$$(\mathsf{Q} \times \mathsf{R})_{(a,c),(b,d)} = \mathsf{Q}_{a,b} \otimes \mathsf{R}_{c,d}.$$

If the automata are positive weighted then $\varepsilon_{A\times C} = (\varepsilon_A, \varepsilon_C)$, $\varepsilon_{B\times D} = (\varepsilon_B, \varepsilon_D)$.

This just says that the weight of a transition from (q, r) to (q', r') with left signal (a, c) and right signal (b, d) is the product of the weights of the transition $q \to q'$ with signals a and b, and the weight of the transition $r \to r'$ with signals c and d. The following simple lemma was proved in [4].

Lemma 3.5 *If $\mathbf{Q}$ and $\mathbf{R}$ are positive weighted automata then so is $\mathbf{Q} \times \mathbf{R}$ and*

$$\mathbf{N}(\mathbf{Q} \times \mathbf{R}) = \mathbf{N}(\mathbf{Q}) \times \mathbf{N}(\mathbf{R}).$$

Hence if $\mathbf{Q}$ and $\mathbf{R}$ are Markov automata then so is $\mathbf{Q} \times \mathbf{R}$.

Lemma 3.6 *If $\mathbf{Q}$ and $\mathbf{R}$ are Markov automata then $(\mathbf{Q} \times \mathbf{R})^k = \mathbf{Q}^k \times \mathbf{R}^k$.*

3.2.2 Parallel with communication

Definition 3.7 *Given weighted automata* $\mathbf{Q}^X_{Y;A,B}$ *and* $\mathbf{R}^Z_{W;B,C}$ *the* communicating parallel composite of weighted automata $\mathbf{Q}||\mathbf{R}$ *has set of states* $Q \times R$*, left interfaces* A*, right interface* C*,in and out interfaces* $X \times Z$*,* $Y \times W$*, sequential interfaces* $\gamma_{0,\mathbf{Q}} \times \gamma_{0,\mathbf{R}}$*,* $\gamma_{1,\mathbf{Q}} \times \gamma_{1,\mathbf{R}}$ *and transition matrices*

$$(\mathsf{Q}||\mathsf{R})_{a,c} = \sum_{b \in B} \mathsf{Q}_{a,b} \otimes \mathsf{R}_{b,c}.$$

The following simple lemmas were proved in [4].

Lemma 3.8 *If* $\mathbf{Q}$ *and* $\mathbf{R}$ *are positive weighted automata then so is* $\mathbf{Q}||\mathbf{R}$ *and* $\mathbf{N}(\mathbf{N}(\mathbf{Q})||\mathbf{N}(\mathbf{R})) = \mathbf{N}(\mathbf{Q}||\mathbf{R})$.

3.2.3 Parallel constants

Definition 3.9 *Given a relation* $\rho \subseteq A \times B$ *we define a weighted automaton* $\mathbf{Par}(\rho)$ *as follows: it has one state* $*$ *say. Top and bottom interfaces have one element. The transition matrices* $[\mathsf{Par}(\rho)_{a,b}]$ *are* 1×1 *matrices, that is, real numbers. Then* $\mathsf{Par}(\rho)_{a,b} = 1$ *if* ρ *relates* a *and* b*, and* $\mathsf{Par}(\rho)_{a,b} = 0$ *otherwise. If* $(\varepsilon_A, \varepsilon_B) \in \rho$ *then* $Par(\rho)$ *is also positive weighted.*

Parallel connectors Some special cases, all described in [8], have particular importance and are called *parallel connectors* or *wires*: (i) the automaton corresponding to the identity function 1_A, considered as a relation on $A \times A$ is also called 1_A; (ii) the automaton corresponding to the diagonal function $\Delta : A \to A \times A$ (considered as a relation) is called Δ_A; the automaton corresponding to the opposite relation of Δ is called Δ^o_A; (iii) the automaton corresponding to the function $twist : A \times B \to B \times A$ is again called $twist_{A,B}$; (iv) the automaton corresponding to the projection function $A \to \{*\}$ is called p and its opposite p^o.

The role of parallel wires is to *equate signals.*

Parallel codiagonal The automaton corresponding to the function $\nabla : A + A \to A$ is called the *parallel codiagonal*, and is denoted ∇ where is no confusion. The automaton corresponding to the opposite relation is written ∇^o.

3.3 Some derived operations

3.3.1 Local sum

Given weighted automata $\mathbf{Q}^X_{Y;A,B}$ and $\mathbf{R}^Z_{W;A,B}$ the local sum $\mathbf{Q} + \mathbf{R}$ is defined to be $\nabla^o_A||(\mathbf{Q} \boxplus \mathbf{R})||\nabla_A$. It has top and bottom interface $X + Z$ and $Y + W$, and left and right interfaces A and B.

3.3.2 Local sequential composition

Given weighted automata $\mathbf{Q}^X_{Y;A,B}$ and $\mathbf{R}^Y_{Z;A,B}$ the local sequential composite $\mathbf{Q} \bullet \mathbf{R}$ is defined to be $\nabla^o_A||(\mathbf{Q} \circ \mathbf{R})||\nabla_A$. It has top and bottom interface X and Z, and left and right interfaces A and B.

3.3.3 Sequential feedback

Given weighted automata $\mathbf{Q}^{X+Z}_{Y+Z;A,B}$ sequential feedback $\mathsf{Sfb}_Z(\mathbf{Q}^{X+Z}_{Y+Z;A,B})$ is defined to be

$$(1_X \boxplus i_Z) \circ (1_X \boxplus \nabla^o_Z) \circ (\mathbf{Q} \boxplus 1_Z) \circ (1_Y \boxplus \nabla_Z) \circ (1_Y \boxplus i^o_Z).$$

This formula is easier to understand looking ahead at the graphical representation in the next section.

3.3.4 Parallel feedback

Given weighted automata $\mathbf{Q}^X_{Y;A\times C,B\times C}$ parallel feedback $\mathsf{Pfb}_C(\mathbf{Q}^X_{Y;A+C,B+C})$ is defined to be

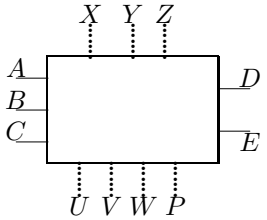
$$(1_A \times p^o_C)||(1_A \times \Delta_C)||(\mathbf{Q} \times 1_C)||(1_B \times \Delta^o_C)||(1_B \times p_C).$$

This formula is also easier to understand looking ahead to the graphical representation in the next section.

3.4 Graphical representation of expressions of weighted automata

Not only do weighted automata have a graphical representation, as seen above, but so also do expressions of automata, as described in [8]. We extend the representation given in that paper to the combination of sequential and parallel operations and constants. We will see in section 4.1 that the graphical representation of single automata is actually a special case of this new representation of expressions, modulo the equations satisfied by wires (the Frobenius and separable algebra equations first introduces in [3], see also [13]).

In general an expression will be represented by a diagram of the following sort:

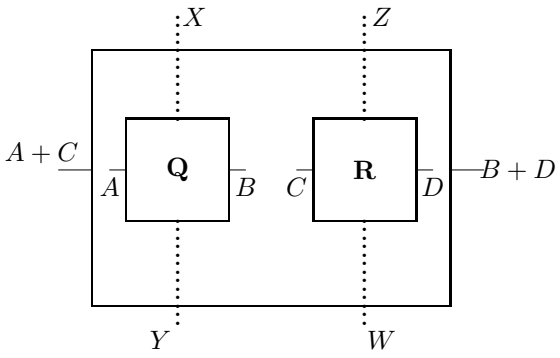


The multiple lines on the left and right hand sides correspond to parallel interfaces which are products of sets. For example, the component has left interface $A\times B\times C$. Instead the multiple lines on the top and bottom correspond to sequential interfaces which are disjoint sums of sets, so the top interface is $X + Y + Z$.

3.5 Operations and constants

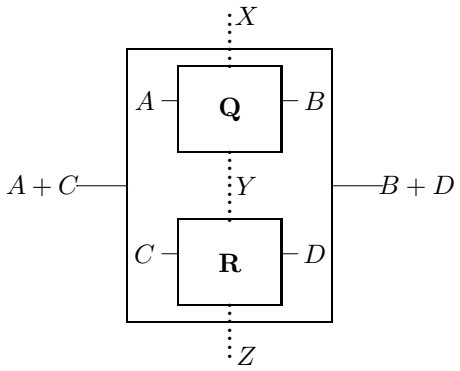
3.5.1 Sum

The sum $\mathbf{Q}^X_{Y;A,B} \boxplus \mathbf{R}^Z_{W;C,D}$ is represented as:



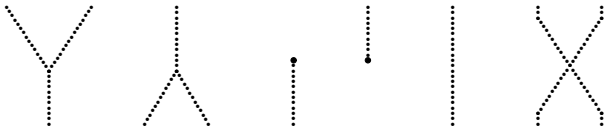
3.5.2 Sequential composition

The sequential composite $\mathbf{Q}^X_{Y;A,B} \circ \mathbf{R}^Y_{Z;C,D}$ is represented as:



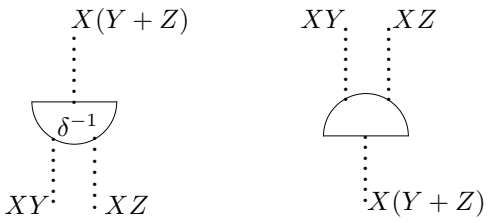
3.5.3 Sequential connectors

The various sequential connectors are represented as:



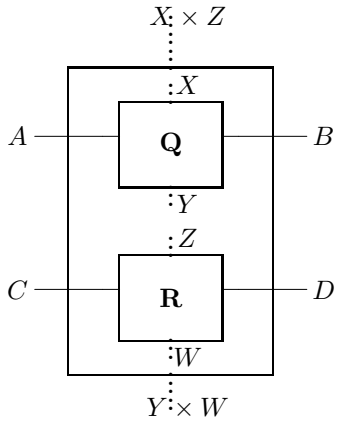
3.5.4 Distributive law

The distributive law $\delta^{-1} : X \times (Y + Z) \to X \times Y + X \times Z$ and its opposite are represented as:



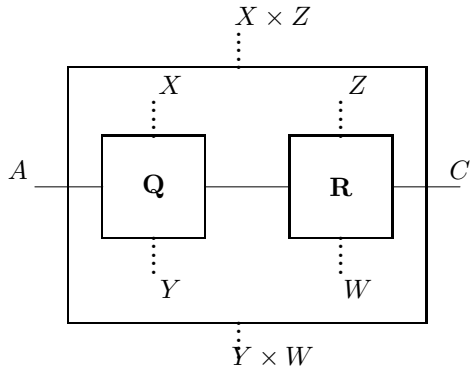
3.5.5 Product

The product $\mathbf{Q}^X_{Y;A,B} \times \mathbf{R}^Z_{W;C,D}$ is represented as:



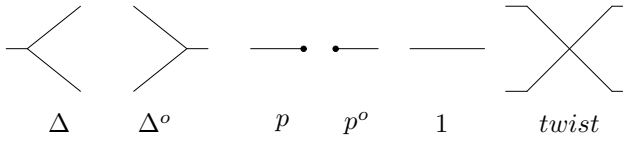
3.5.6 Parallel with communication

The parallel with communication $\mathbf{Q}^X_{Y;A,B} || \mathbf{R}^Z_{W;B,C}$ is represented as:



3.5.7 Parallel connectors

The various parallel connectors are represented as:



3.5.8 Parallel codiagonal

The parallel codiagonal $\nabla : A + A \to A$ and its opposite ∇^o are represented as:

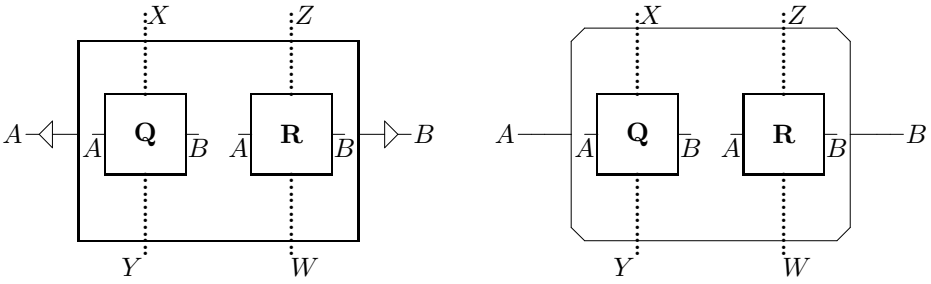
3.6 Some derived operations and constants

3.6.1 Repeated sequential and repeated parallel operations

When representing repeated sequential operations, frequently we omit all but the last bounding rectangle. With care this does not lead to ambiguity - different interpretations lead to at worst isomorphic automata. We do the same with repeated parallel operations. It is not possible to removed bounding rectangles for mixed repeated and parallel operations without serious ambiguity.

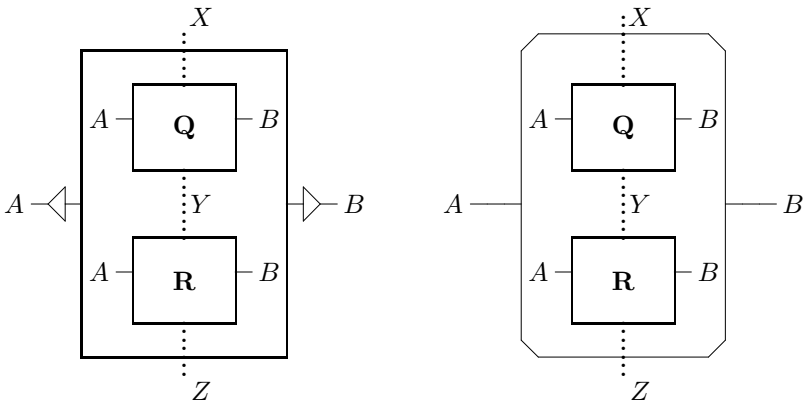
3.6.2 Local Sum

The local sum $\mathbf{Q}^X_{Y;A,B} + \mathbf{R}^Z_{W;A,B}$ is represented by the first diagram below which includes the parallel codiagonals, but also, since it is such a common derived operation, more briefly by the second diagram below:



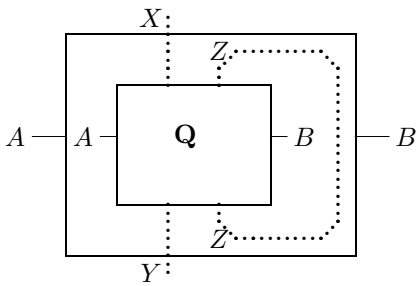
3.6.3 Local sequential

The local sequential $\mathbf{Q}^{X}_{Y;A,B} \bullet \mathbf{R}^{Y}_{Z;A,B}$ is represented by the first diagram below which includes the parallel codiagonals, but also, since it is such a common derived operation, more briefly by the second diagram below:



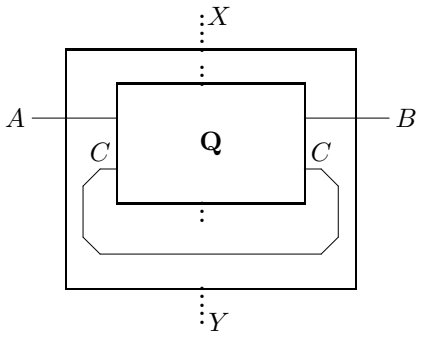
3.6.4 Sequential feedback

The sequential feedback $\mathsf{Sfb}_Z(\mathbf{Q}^{X+Z}_{Y+Z;A,B})$ is represented by the diagram:



3.6.5 Parallel feedback

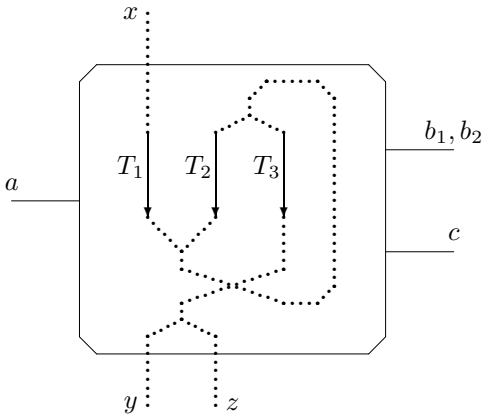
The sequential feedback $\mathsf{Pfb}_C(\mathbf{Q}^X_{Y;A\times C,B\times C})$ is represented by the diagram:



4 Examples

4.1 Any automaton is in $\Sigma(E)$

If E is the set of automata with two states with a single transition, then any automaton may be given as a sequential expression of elements in E. We illustrate by considering the first example of section 2. Let T_1, T_2, T_3 be the three single transition automata as follows: T_1 has the single transition labelled on the left by a and the right by (b_1, c) with weight 2, T_2 has the single transition labelled on the left by a and the right by (b_1, c) with weight 3. and T_3 has the single transition labelled on the left by a and the right by (b_2, c) with weight 1, Then the following diagram shows how the automaton may be given as $T_1 \boxplus T_2 \boxplus T_3$ composed sequentially with sequential wires:



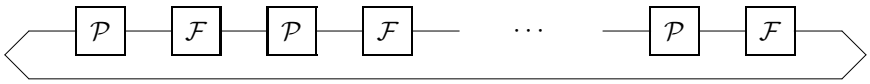
4.2 The dining philosophers system

The model of the dining philosophers problem we consider is an expression in the algebra, involving also the automata **Phil** and **Fork**. The system of n dining philosophers is

$$\mathbf{DF}_n = \mathsf{Pfb}(\mathbf{Phil}||\mathbf{Fork}||\mathbf{Phil}||\mathbf{Fork}||\cdots||\mathbf{Phil}||\mathbf{Fork}),$$

where in this expression there are n philosophers and n forks.

The system is represented by the following diagram, where we abbreviate **Phil** to $\mathcal{P}$ and **Fork** to $\mathcal{F}$.



Let us examine the case when $n = 2$ with initial state $(1, 1, 1, 1)$. Let $\mathbf{Q}$ be the reachable part of $\mathbf{DF}_2$. The states reachable from the initial state are $q_1 = (1, 1, 1, 1)$, $q_2 = (1, 3, 3, 2)$, $q_3 = (3, 2, 1, 3)$, $q_4 = (1, 1, 4, 2)$, $q_5 = (4, 2, 1, 1)$, $q_6 = (1, 3, 2, 1)$, $q_7 = (2, 1, 1, 3)$, $q_8 = (2, 3, 2, 3)$ (q_8 is the unique deadlock state). The single matrix of the automaton Q, using this ordering of the states, is

$$\begin{bmatrix}
\frac{1}{4} & 0 & 0 & 0 & 0 & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \\
0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 \\
0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 \\
\frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 \\
\frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 \\
0 & \frac{1}{3} & 0 & 0 & 0 & \frac{1}{3} & 0 & \frac{1}{3} \\
0 & 0 & \frac{1}{3} & 0 & 0 & 0 & \frac{1}{3} & \frac{1}{3} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}.$$

Calculating powers of this matrix we see that the probability of reaching deadlock from the initial state in 2 steps is $\frac{23}{48}$, in 3 steps is $\frac{341}{576}$, and in 4 steps is $\frac{4415}{6912}$.

4.2.1 The probability of deadlock

We describe here the result of [4] in which we apply Perron-Frobenius theory to the problem of deadlock in the Dining Philosopher problem..

Definition 4.1 *Consider a Markov automaton* $\mathbf{Q}$ *with input and output sets being one element sets* $\{\varepsilon\}$. *A state* q *is called a* deadlock *if the only transition out of* q *with positive probability is a transition from* q *to* q *(the probability of the transition must necessarily be* 1*).*

Proposition 4.2 *(Perron-Frobenius) Consider a Markov automaton* $\mathbf{Q}$ *with interfaces being one element sets, with an initial state* q_0. *Suppose that (i)* $\mathbf{Q}$ *has precisely one reachable deadlock state, (ii) for each reachable state, not a deadlock, there is a path with non-zero probability to* q_0*, and (iii) for each reachable state* q *there is a transition with non-zero probability to itself.*
Then the probability of reaching a deadlock from the initial state in k *steps tends to* 1 *as* k *tends to infinity.*

Corollary 4.3 *In the dining philosopher problem* $\mathbf{DF}_n$ *with* q_0 *being the state* $(1, 1, \cdots, 1)$ *the unique reachable deadlock is* $(3, 2, 3, 2, \cdots, 3, 2)$. *The initial state is reachable from all other reachable states. Hence the probability of reaching a deadlock from the initial state in* k *steps tends to* 1 *as* k *tends to infinity.*

Remark. The corollary does not depend on the specific positive probabilities of the actions of the philosophers and forks. Hence the result is true with any positive probabilities. In fact, different philosophers and forks may have different probabilities without affecting the conclusion of the corollary.

5 Sofia's birthday party

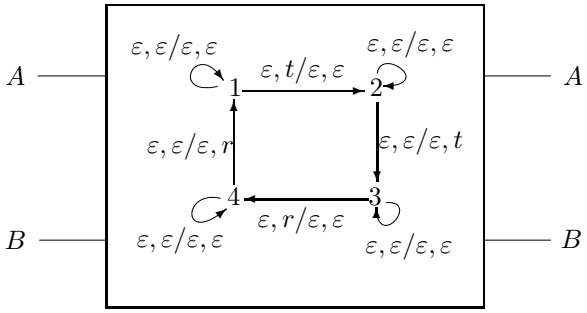
The example we would like to describe is a variant of the Dining Philosopher Problem which we call Sofia's Birthday Party. Instead of a circle of philosophers around a table with as many forks, we consider a circle of seats around a table separated by forks on the table. Then there are a number of children (not greater than the number of seats). The protocol of each child is the same as that of a philosopher. However in addition, if a child is not holding a fork, and the seat to the right is empty, the child may change seats - the food may be better there.

To simplify the problem we will assume that all transitions have weight 0 *or* 1, *so the transitions of components we mention will all have weight*1.

To describe this we need six automata – a child $\mathcal{C}$, an empty seat $\mathcal{E}$, a fork $\mathcal{F}$, two transition elements $\mathcal{L}$ and $\mathcal{R}$, and the identity 1_A of A (a wire). The interface sets involved are $A = \{x, \varepsilon\}$ and $B = \{\varepsilon, t, r\}$.

The transition elements have left and right interfaces $A \times B$. The graph of the of the transition element $\mathcal{L}$ has two vertices p and q and one labelled edges $x, \varepsilon/\varepsilon, \varepsilon : q \to p$. Its top interface is $Q = \{q\}$, and its bottom interface is $P = \{p\}$. The graph of the transition element $\mathcal{R}$ also has two vertices p and q, and has one labelled edges $\varepsilon, \varepsilon/x, \varepsilon : q \to p$. Its top interface is also $Q = \{q\}$, and its bottom interface is $P = \{p\}$. The empty seat $\mathcal{E}$ has left and right interfaces $A \times B$. The graph of the empty seat has one vertex e and one labelled edge $\varepsilon, \varepsilon/\varepsilon, \varepsilon : e \to e$. Its top interface is P and its bottom interface is Q. The functions γ_0, γ_1 are uniquely defined.

The child $\mathcal{C}$ has labelled graph as follows:



The states have the following interpretation: in state 1 the child has no forks; in state 2 it has a fork in its left hand; in state 3 it has both forks (and can eat); in state 4 it has returned it left fork. The child's top interface is P and its

bottom interface is Q. The function γ_0 takes p to 1; the function γ_1 takes q to 1.

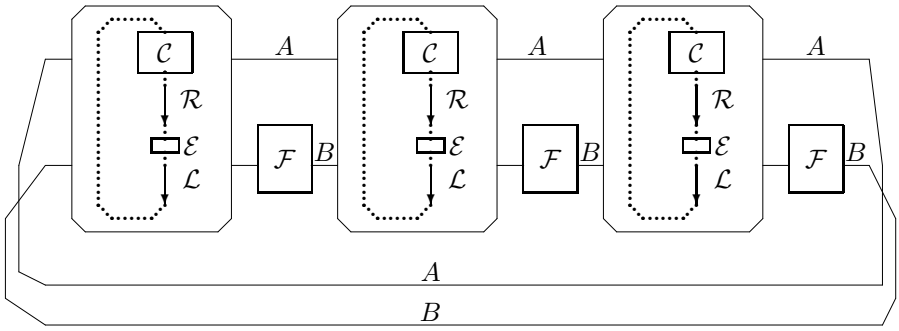
The fork $\mathcal{F}$ is as in the dining philosopher system (but with all transtions weighted 1).

Let $\mathcal{S} = \mathsf{Cfb}_P(C \bullet R \bullet E \bullet L)$. This automaton has the following interpretation – it can either be a child (on a seat) or an empty seat. The transition elements $\mathcal{R}$ and $\mathcal{L}$ allow the seat to become occupied or vacated. It is straightforward to see that this automaton is a positive weighted automaton.

Then Sofia's Birthday Party is given by the normalization of expression

$$\mathsf{Pfb}_{A\times B}(\mathcal{S}||(1_A \times \mathcal{F})||\mathcal{S}||(1_A \times \mathcal{F})||...||\mathcal{S}||(1_X \times \mathcal{F})).$$

This automaton has the behaviour as informally described above. Its diagrammatic representation is:



Notice that though Sofia's Birthday Party belongs to $\Pi\Sigma\Sigma(E)$ slight variations of the system belong instead to $\Pi\Sigma\Pi\Sigma(E)$, for example the system where more than one child may occupy a seat (communicating there). If the system starts in a state with as many children as seats - then movement is impossible and the system is equivalent to the dining philosophers.

Let us look at a particular case of Sofia's Birthday Party in more detail. Consider the system with three seats, and two children. There are 36 states reachable from initial state $(5, 1, 1, 1, 1, 1)$, where 5 is the state in which the seat is empty, and there are 141 transitions.

The states are

$(5,1,1,1,1,1)$	$(5,1,1,3,2,1)$	$(5,3,2,1,1,1)$	$(5,3,2,3,2,1)$
$(1,1,1,1,5,1)$	$(1,3,2,1,5,1)$	$(5,1,1,3,3,2)$	$(5,3,2,3,3,2)$
$(5,3,3,2,1,1)$	$(1,3,3,2,5,1)$	$(1,1,5,1,1,1)$	$(2,1,1,1,5,3)$
$(2,1,5,1,1,3)$	$(2,3,2,1,5,3)$	$(2,3,3,2,5,3)$	$(5,1,1,1,4,2)$
$(5,3,2,1,4,2)$	$(5,1,4,2,1,1)$	$(1,1,4,2,5,1)$	$(2,1,4,2,5,3)$
$(1,1,5,3,2,1)$	$(2,1,5,3,2,3)$	$(3,2,1,1,5,3)$	$(3,2,5,1,1,3)$
$(3,2,5,3,2,3)$	$(5,3,3,2,4,2)$	$(3,2,4,2,5,3)$	$(1,1,5,3,3,2)$
$(4,2,1,1,5,1)$	$(4,2,5,1,1,1)$	$(4,2,5,3,2,1)$	$(5,1,4,2,4,2)$
$(4,2,4,2,5,1)$	$(1,1,5,1,4,2)$	$(4,2,5,3,3,2)$	$(4,2,5,1,4,2)$.

Then in 12 of these states there is a child eating: only one may eat at a time. It is straightforward to calculate the 36×36 Markov matrix and iterating show that the probability of a child eating after 1 step from initial state $(5, 1, 1, 1, 1, 1)$ is 0, after 2 steps is $\frac{19}{60}$, after 3 steps is $\frac{98}{225}$, after 4 steps is $\frac{49133}{108000}$, after 5 steps is $\frac{1473023}{3240000}$ and after 100 steps is 0.3768058221.

References

[1] C. Baier, M. Grösser, F. Ciesinski, *Model checking linear-time properties of probabilistic systems*, Monographs in Theoretical Computer Science, 519-596, Springer, 2009.

[2] R. Blute, A. Edalat, P. Panangaden, *Bisimulation for Labelled Markov Processes*, Proceedings of the 12th Annual IEEE Symposium on Logic in Computer Science, pp. 95-106, 1997.

[3] A. Carboni, R.F.C. Walters, *Cartesian bicategories I*, Journal of Pure and Applied Algebra, 49, 11–32, 1987.

[4] L. de Francesco Albasini, N. Sabadini, R.F.C. Walters, *The compositional construction of Markov processes*, to appear in Applied Categorical Structures, (also arXiv:0901.2434).

[5] L. de Francesco Albasini, N. Sabadini, R.F.C. Walters, *Cospans and spans of graphs: a categorical algebra for the sequential and parallel composition of discrete systems*, arXiv:0909.4136.

[6] M. Droste, W. Kuich, H. Vogler, (Eds.), *Handbook of Weighted Automata*, EATCS Monographs in Theoretical Computer Science :2009.

[7] J. Hillston, *A Compositional Approach to Performance Modelling*, Cambridge University Press, 1996.

[8] P. Katis, N. Sabadini, R.F.C. Walters, *Span(Graph): A categorical algebra of transition systems*, Proc. AMAST '97, SLNCS 1349, pp 307–321, Springer Verlag, 1997.

[9] P. Katis, N. Sabadini, R.F.C. Walters, *A formalisation of the IWIM Model.* in: Proc. COORDINATION 2000,(Eds.) Porto A., Roman G.-C., LNCS 1906:267-283, Springer Verlag, 2000.

[10] Nancy A. Lynch, Roberto Segala, Frits W. Vaandrager, *Compositionality for Probabilistic Automata*, Proc. CONCUR 2003, Springer Lecture Notes in Computer Science, 2761, pp 204-222, 2003.

[11] Mehryar Mohri, *Weighted automata algorithms*, In Manfred Droste, Werner Kuich, and Heiko Vogler, editors, Handbook of Weighted Automata, Monographs in Theoretical Computer Science, pages 213-254, Springer, 2009.

[12] M. O Rabin, *Probabilistic Automata*, Information and Control 6, pp.230-245, 1963.

[13] R. Rosebrugh, N. Sabadini, R.F.C. Walters, *Generic commutative separable algebras and cospans of graphs*, Theory and Applications of Categories, 15, 264-177, 2005.

[14] A. Sokolova, E.P. de Vink, *Probabilistic automata: system types, parallel composition and comparison*, Springer Lecture Notes in Computer Science, 2925, pp. 1-43, 2004.