Fundamenta Informaticae 171 (2020) 221–237
DOI 10.3233/FI-2020-1880
IOS Press

CospanSpan(Graph): a Compositional Description of the Heart System

Alessandro Gianola*
Faculty of Computer Science
Free University of Bozen-Bolzano, Italy
gianola@inf.unibz.it

Stefano Kasangian
Dipartimento di Matematica
Università degli studi di Milano, Italy
stefano.kasangian@unimi.it

Desiree Manicardi
Dipartimento di Scienza ed Alta Tecnologia
Università degli Studi dell'Insubria, Italy
desiree.manicardi@gmail.com

Nicoletta Sabadini
Dipartimento di Scienza ed Alta Tecnologia
Università degli Studi dell'Insubria, Italy
nicoletta.sabadini@uninsubria.it

Filippo Schiavio
Faculty of Informatics
Università della Svizzera Italiana, Switzerland
filippo.schiavio@usi.ch

Simone Tini
Dipartimento di Scienza ed Alta Tecnologia
Università degli Studi dell'Insubria, Italy
simone.tini@uninsubria.it

Abstract. In this paper, we recall the basic features of the CospanSpan(Graph) algebra for the compositional description of reconfigurable hierarchical networks. In particular, we focus on compositionality and on the possibility of describing the interactions among physical/biological systems, using a parallel with communication operation not considered in the usual Kleene's algebra. As a novel application, we give a complete compositional description in Span(Graph) of a simplified version of the heart system.

Keywords: Automata, Categories, Composition, Biological Systems

*Address for correspondence: Faculty of Computer Science, Free University of Bozen-Bolzano, Italy

1. Introduction

Historically, automata had a fundamental role both in technical-scientific and in philosophic-literary areas. From ancient times, in fact, we have examples of mechanical devices simulating some of the aspects of living beings, from movement to speech: let us recall, for example, the wonderful automata of Vaucanson and Jaquet-Droz (the duck, the flute player...) or Von Kempelen's speaking machine, unfortunately less known than the (in)famous chess player "the Turk". At the beginning of the 20th century the fundamental results of mathematicians like Alan Turing provided a solid foundation for Automata Theory and gave rise to a paradigmatic shift from a purely technical-engineering point of view of automata/machines as mechanical devices towards their mathematical formalization. That fundamental step, opening the way to Cybernetics and Artificial Intelligence, reconciles their technical, cognitive and symbolic aspects. Indeed Turing Machines became the standard model to formalize, for the first time in history, the concepts of sequential machine and of "mechanically" solvable and non-solvable problems. In the same period, starting from the seminal work done by McCulloch and Pitts on modeling neural networks using finite state automata and from Von Neumann cellular automata, the idea of *automata networks* became fundamental. Nowadays, this idea is pervasive and dominant in computer science as well as in our society, and automata had evolved from simple mechanisms to complex structures consisting of *networks of interacting and interconnected entities*, with variable topology, that can even exhibit "cognitive" capabilities, as calculus, speech and visual recognition, learning and self-organization. Indeed the distinction between artificial and living entities seems to become "fuzzy". So, it is very natural to consider Automata Theory and Biology as very close disciplines, with a long and very fruitful tradition of reciprocal influences, from robotics to biology-inspired models of computation ([1, 2, 3]). Unfortunately, whereas we could be reasonable satisfied with Turing Machines (and finite state automata) when dealing with discrete, isolated and sequential computation devices, we do not have a general model for automata networks that could play an analogous role. A variety of models, and different mathematical approaches, have been proposed in literature to formalize "complex" systems or networks.

In this paper, we recall the CospanSpan(Graph) categorical model proposed by Katis-Sabadini-Walters [4, 5], and, in particular, the *parallel with communication operation*, that allows two components to *simultaneously* evolve through parallel communication, likewise two connected gears. This operation is clearly fundamental when physical/biological devices are under consideration. The operations of parallel with communication and non-sequential feedback were not considered by Kleene and Turing, but are fundamental in modeling complex systems/processes as variable topology networks of interacting and cooperating automata. We observe that this algebraic framework gives us immediately *the possibility of a compositional and hierarchical description of networks* [6]. This algebra also provides a calculus of connectors that allows an elegant description of a signal broadcasting. The original algebra can be extended introducing probability and time [7, 8]. In this paper we would like to focus on the importance of an algebraic approach for the *compositional* description of variable topology networks that leads to a natural formalization of biological systems. In particular, as a *genuinely novel* application, we present a Span(Graph) version of the heart system. Here, we assume a normal functionality of the heart, but the model could be easily extended to cope with abnormal behaviours

like cardiac arrhythmia or other cardiac pathologies. The synergy between heart and pacemaker could also be modeled, following [9].

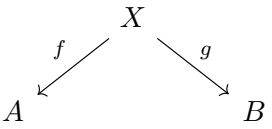
2. CospanSpan(Graph): an algebraic formalism for automata networks

In this section we recall the algebra $\mathbf{Span}(\mathbf{Graph})$ and its dual counterpart $\mathbf{Cospan}(\mathbf{Graph})$, as introduced by Benabou in [10]. In the following we will show that CospanSpan(C) becomes a natural syntax for a very precise semantic model when $C := Graph$: in fact, the operations of the algebra correspond in a natural way to operations on automata with interfaces (and their connectors) that extend Kleene's algebras.

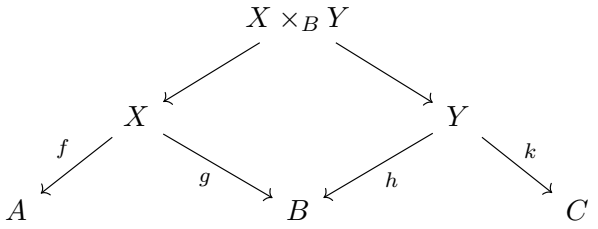
2.1. The algebra of spans

2.1.1. Span(C): an abstract view

Definition 2.1. Given a category $\mathbf{C}$ with finite limits, we define a new category $\mathbf{Span}(\mathbf{C})$ describing its objects and arrows. Objects of $\mathbf{Span}(\mathbf{C})$ are the same objects of $\mathbf{C}$; arrows of $\mathbf{Span}(\mathbf{C})$ from A to B (with A, B objects) are spans, that is pairs of arrows $(f : X \to A, g : X \to B)$ of $\mathbf{C}$ with common domain, often written:

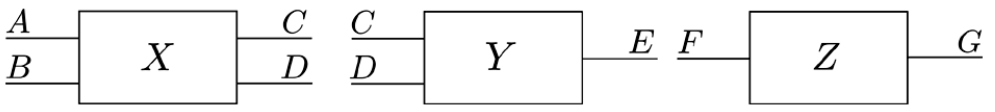


Composition of spans $(f : X \to A, g : X \to B)$ and $(h : Y \to B, k : Y \to C)$ is by pullback (restricted product):

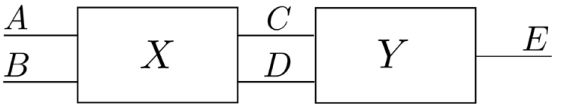


A span $(f : X \to A, g : X \to B)$ will also be written $X : A \to B$, and the composition of span will be indicated with the notation $X \cdot Y : A \longrightarrow C$. The identity span of object A is $(1_A, 1_A)$. The category $\mathbf{Span}(\mathbf{C})$ is actually symmetric monoidal with the *tensor* product of two spans $(f : X \to A, g : X \to B)$ and $(h : Y \to C, k : Y \to D)$ being $(f \times h : X \times Y \to A \times C, g \times k : X \times Y \to B \times D)$, denoted by $X \times Y$ or $X \otimes Y$. The identity span of object A is $(1_A, 1_A)$.

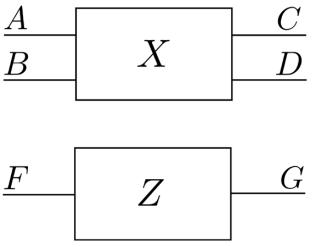
In [4] an informal geometric description (in the style of monoidal category string diagrams) was introduced for the operations in $\mathbf{Span}(\mathbf{C})$. For example, spans $(X \to A \times B, X \to C \times D)$, $(Y \to C \times D, Y \to E)$ and $(Z \to F, X \to G)$ are represented by pictures of *components with ports*:



Then the composite of the first two spans is pictured as:



while the tensor of the first and third is pictured as:



In addition there are constants of the algebra which are pictured as operation on wires which enable the depiction of fanning out of wires and feedback, and hence of general circuit diagrams. We give some examples of constants in $\mathbf{Span}(\mathbf{C})$ (that are described also in [4, 11]).

- The identity span $1_X : X \longrightarrow X$ has head X and two legs $1_X, 1_X$. It is denoted by a plain wire.



- The span with head X and legs $1_X : X \longrightarrow X$, $\Delta_X : X \longrightarrow X \times X$ is called the diagonal of X and is denoted also $\Delta : X \longrightarrow X \times X$. The span with head X and legs $\Delta_X : X \longrightarrow X \times X$,$1_X : X \longrightarrow X$ is called the reverse diagonal, and is denoted $\nabla : X \times X \longrightarrow X$

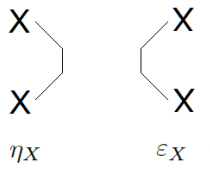


Diagonal Reverse diagonal

- The span with head $X \times Y$ and legs $1_{X\times Y} : X \times Y \longrightarrow X \times Y$, $p_X : X \times Y \longrightarrow X$ is denoted p_X, is called a projection and is pictured by the termination of the wire Y. There is also a similarly defined reverse projection denoted p_X^*.

- Consider the terminal object, denoted I. The span with head X and legs $! : X \longrightarrow I$, $\Delta_X : X \longrightarrow X \times X$ is called η_X. The span with head X and legs $\Delta_X : X \longrightarrow X \times X$, $! : X \longrightarrow I$

is called ε_X The two spans are pictured in the following figure:

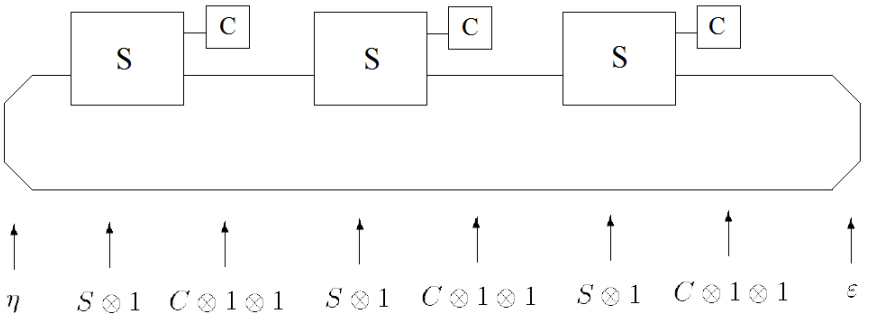


The correspondence between on the one hand constants and operations and on the other hand their geometric representations results in the fact that expressions in the algebra have corresponding circuit or system diagrams. This is clarified in the following example.

Example 2.2. Given spans $S : X \longrightarrow X \times X$ and $C : X \longrightarrow I$, the expression

$$\eta_X \cdot (S \otimes 1_X) \cdot (C \otimes 1_X \otimes 1_X) \cdot (S \otimes 1_X) \cdot (C \otimes 1_X \otimes 1_X) \cdot (S \otimes 1_X) \cdot (C \otimes 1_X \otimes 1_X) \cdot \varepsilon_X$$

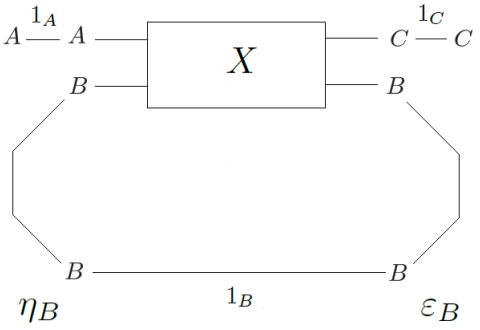
has system diagram (which graphically forms a feedback):



As shown in the previous example, η and ε permit a natural feedback operation. More formally, in $\mathbf{Span(C)}$ it is possible to abstractly define the notion of *parallel feedback* [11]:

Definition 2.3. Given a span $X : A \times B \longrightarrow C \times B$, we call *(abstract) parallel feedback* of X with respect to B, denoted by $AbPfb_B(X)$, the span denoted by the following algebraic expression:

$$(1_A \otimes \eta_B) \cdot (X \otimes 1_B) \cdot (1_C \otimes \varepsilon_B)$$



Notice that the diagrammatic representation of $AbPfb_B(X)$ involves joining the right interface B to the left interface B.

2.1.2. $\mathbf{Span}(\mathbf{Graph})$, a parallel algebra of automata

Surprisingly, $\mathbf{Span}(\mathbf{C})$, when $\mathbf{C}$ is the category of (finite) directed graphs ($\mathbf{Span}(\mathbf{Graph})$), provides a very natural mathematical framework for describing the composition of *automata with interfaces* (or communication ports, as in circuit theory). Consider a span of graphs $(\delta_0 : X \to A, \delta_1 : X \to B)$. The graph X may be considered as the graph of states and transitions of an automata with interfaces, and it is called the *head* of the span. The graph A is the graph of states and transitions of the combined left ports and B is the graph of states and transitions of the combined right ports. The graph morphism δ_0 associates to a state and to a transition of the automaton X the corresponding state and transition of the left ports A; the morphism δ_1 does the same for the right ports.

For all the examples of this paper the left and right ports have only one state so that we tend to ignore that; then δ_0 and δ_1 are a double labelling of the transitions of the automaton X by transitions on the left ports and transitions on the right ports. More intuitively each transition of the component has an effect on all its interfaces, maybe the null effect ε.

In the case that the left and right ports have one state, the operations of composition and tensor of spans have a simple description in terms of operations on automata. The tensor of two automata has states being pairs of states, one of each automata, and has as transitions pairs of transitions between the corresponding pairs of states. The composition of automata has similarly states being pairs of states, and transitions being pairs of transitions but *only those pairs of transitions whose labels on the connected ports are the same*. In the following, we will call the span composition *parallel composition with communication* and the tensor *parallel composition without communication*.

Remark 2.4. If one wants a parallel algebra of automata which also includes initial and final states of the automata, a slight variation of the above is possible: instead of the category of graphs take the category $(I + J)\backslash\mathbf{Graph}$ whose objects are graph morphisms $(I + J) \to X$, that is consist of two graph morphisms $I \to X$ and $J \to X$, to be thought of as the initial states and final states of X. Then the parallel algebra is $\mathbf{Span}((I + J)\backslash\mathbf{Graph})$.

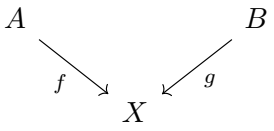
Remark 2.5. In [8] *timed actions* with different duration have been considered. Composition is obtained by considering a linear (w.r.t. transitions) number of extra "internal" states. The intended meaning is that a component that interacts with a "faster" one could be still doing an action (hence being in an internal state) when the other one has completed the transition. We give a simple example:

Example 2.6. (Two actions which synchronize with an arbitrary duration)
Consider two automata $\mathcal{G}_1$, $\mathcal{G}_2$ and two "non-atomic" actions, one of $\mathcal{G}_1$, one of $\mathcal{G}_2$. The action of $\mathcal{G}_1$ is $\{a/b : 0 \to 1, \varepsilon/\varepsilon : 1 \to 1, d/e : 1 \to 2\}$; the action of $\mathcal{G}_2$ is $\{b/c : 0 \to 1, \varepsilon/\varepsilon : 1 \to 1, e/f : 1 \to 2\}$. In the restricted product $\mathcal{G}_1 \cdot \mathcal{G}_2$ these actions synchronize but with arbitrary duration. A typical behaviour is $(0,0) - a/c \to (1,1) - \varepsilon/\varepsilon \to (1,1) - \varepsilon/\varepsilon \to ... - \varepsilon/\varepsilon \to (1,1) - d/f \to (2,2)$.

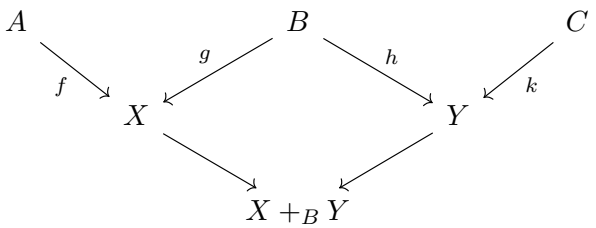
2.2. The algebra of cospans

2.2.1. Cospan(C): an abstract view

Definition 2.7. There is a dual construction $\mathbf{Cospan}(\mathbf{C})$ for categories $\mathbf{C}$ with finite colimits. In fact, $\mathbf{Cospan}(\mathbf{C}) = \mathbf{Span}(\mathbf{C}^{\mathbf{op}})$, but it is better seen describing explicitly its objects and arrows. Objects of $\mathbf{Cospan}(\mathbf{C})$ are the same as objects of $\mathbf{C}$; arrows of $\mathbf{Cospan}(\mathbf{C})$ from A to B are cospans, that is, pairs of arrows $(f : A \to X, g : B \to X)$ of $\mathbf{C}$ with common codomain, also written as:

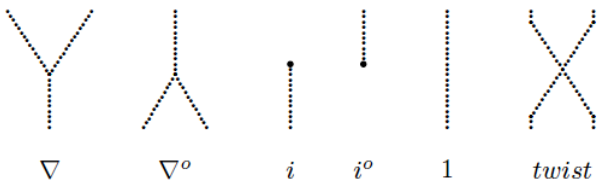


Composition (which is also called *restricted sum*) of $(f : A \to X, g : B \to X)$ and $(h : B \to Y, k : C \to Y)$ is by pushout (glued sum):



A cospan $(f : A \to X, g : B \to X)$ will also be written $X : A \to B$, and the composition of cospan will be indicated with the notation $X + Y : A \longrightarrow C$. The identity cospan of object A is $(1_A, 1_A)$.

Again there are constants of the algebra which are pictured as operation on wires which enable the depiction of joining of wires and sequential feedback. Graphically, we can present these operations as the picture below:



As before, when we will consider $\mathbf{Cospan}(\mathbf{Graph})$ as an algebra of automata where sequential compositions are the main operations, we will use the expression *sequential feedback*.

2.2.2. $\mathbf{Cospan}(\mathbf{Graph})$, a sequential algebra of automata

Analogously, when $\mathbf{C}$ is the the category of (finite) directed graphs, $\mathbf{Cospan}(\mathbf{Graph})$ provides a sequential calculus for automata that generalizes Kleene's one. Consider a cospan of graphs $(\gamma_0 : E \to X, \gamma_1 : F \to X)$. The graph X may be considered as the graph of states and transitions of

an (unlabelled) automaton. The graph E is the graph of *initial* states and transitions and F is the graph of *final* states and transitions. In all the examples considered E and F have only states and not transitions. The graph morphisms γ_0 and γ_1 are often inclusion morphisms of the initial and final states in X.

Remark 2.8. If one wants a sequential algebra of automata which have labelled transitions, a slight variation of the above is possible: instead of the category of graphs take the category $\mathbf{Graph}/(A \times B)$ whose objects are graph morphisms $X \to A \times B$, that is consist of two graph morphisms $X \to A$ and $X \to B$, to be thought of as the left and right labelings of X. Then the sequential algebra is $\mathbf{Cospan}(\mathbf{Graph}/(A \times B))$.

2.3. Cospans and spans of graphs

The two algebras we have described may be combined in a natural way [12, 11]. Consider four graph morphisms $(\delta_0 : X \to A, \delta_1 : X \to B, \gamma_0 : E \to X, \gamma_1 : F \to X)$. From these we may obtain an arrow in the parallel algebra $\mathbf{Span}((E + F)\backslash\mathbf{Graph})$, and also in the sequential algebra $\mathbf{Cospan}(\mathbf{Graph}/(A \times B))$, and hence we may apply both sequential and parallel operations to such automata, obtaining hierarchical nets of automata with evolving geometry. There is a distributive law of parallel composition over sequential, that will be used in the following example. For some details of this see [5].

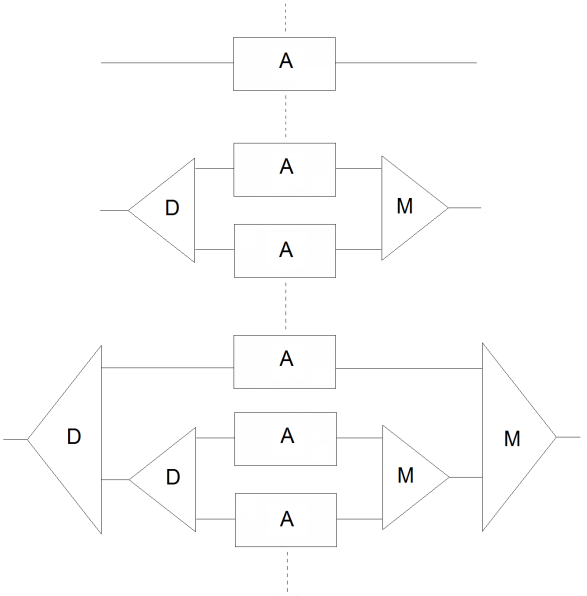
Example 2.9. (Distributed Sort Algorithm)
In [5], it has been described in full details an example of reconfigurable network, that is a Distributed Sort Algorithm: an atomic sort A receives a stream of items to be sorted; if the atomic sort gets full, a new network gets activated in which a divert component D, two atomic sorts A and the merge component M act in parallel. The whole system is the solution of a recursive equation

$$S = A + D \cdot (A \times S) \cdot M$$

which is expanded using the *distributive laws* (between products and the restricted sums) in the following way

$$\begin{aligned} S &= A + D \cdot (A \times S) \cdot M \\ &= A + D \cdot (A \times (A + D \cdot (A \times S) \cdot M)) \cdot M \\ &= A + D \cdot (A \times A) \cdot M + D \cdot (A \times (D \cdot (A \times S) \cdot M)) \cdot M \\ &= \dots \\ &= A + D \cdot (A \times A) \cdot M + D \cdot (A \times (D \cdot (A \times A) \cdot M)) \cdot M + \dots \end{aligned}$$

In this equation the variables are Cospan/Span automata. It gives rise to a network that can be graphically represented as follows:

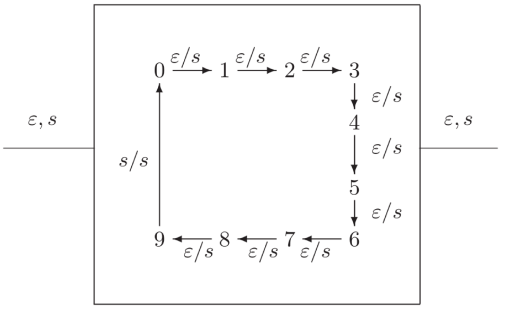


A concrete example of automaton in CospanSpan(Graph) with parallel and sequential feedbacks is given by *Sofia's Birthday Party*, presented in [5, 11] (see in particular [11] for a global representation of the whole automaton).

3. Compositionality

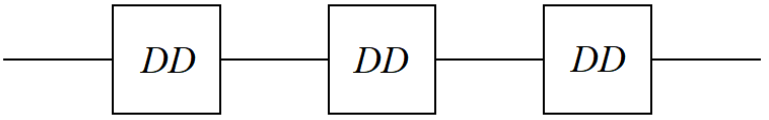
A distinctive feature of CospanSpan(Graph) is *compositionality*. Intuitively, compositionality means that, given a syntactic expression, its global semantics can be deduced *only* by the semantics of its constituents, and this is precisely what our algebraic approach provides. Kleene's algebra, with corresponding automata operations, gives another example of compositional description, for classical *sequential* automata. Various models based on a notion of *product of automata* have been considered in the literature in order to consider the possibility of "parallel communication" among components (for example Zielonka's Asynchronous Automata [13]), but these models are *not* compositional. We argue that in order to fully achieve compositionality, also with respect to parallelism, we will have to abandon both the classical (inherently sequential and closed) model of finite state automata and the well established idea of input/output communication for a new paradigm, considering as fundamental the notion of open systems with communication interfaces. The operations in the algebra CospanSpan(Graph) can be interpreted in a very natural way as operations on automata with interfaces. Hence, it is quite easy to build complex systems starting from simpler components.

Example 3.1. (Decimal Counter)
In the following example ([11]), the left and right ports are both graphs with one state and two transitions ε and s which are displayed on the ports. The automaton has ten states and ten transition (in a circle through the states). The morphisms δ_0 and δ_1 are indicated by doubly-labelling the transitions of the automaton (so, for example, $\delta_0(0 \to 1) = \varepsilon$ and $\delta_1(0 \to 1) = s$).

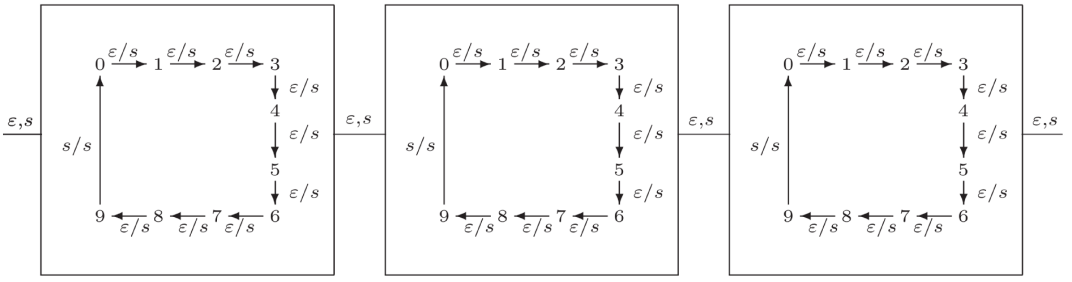


We are actually interested in a variation of this example which we shall call $DecimalDigit$ or more shortly DD which has the same ports but in addition to the ten transitions above also ten loops one on each state, each labelled $(\varepsilon/\varepsilon)$.

Using composition in $\mathbf{Span}(\mathbf{Graph})$ we can form a new automaton $DecimalCounter$ as the span composition of (for simplicity only) three DD's:



or in more detail (though omitting the loops) as:



We notice that a state of $DecimalCounter$ is a triple of states one of each DD, and a transition of $DecimalCounter$ is a triple of simultaneous transitions with the property that the *labels on the connected ports agree*.

For example, starting in state $(0, 1, 8)$ the following is a sequence of transitions of $DecimalCounter$:

$$(0, 1, 8) \xrightarrow{(\varepsilon/s)} (0, 1, 9) \xrightarrow{(\varepsilon/\varepsilon)} (0, 1, 9) \xrightarrow{(\varepsilon/s)} (0, 2, 0)$$

(remember the ε/ε loops on each state of DD) so a transition s on the right-most port increments the counter. In fact, starting from state $(0, 0, 0)$, after 123 transitions labelled s on the right, the state of the network is $(1, 2, 3)$.

Notice that the transition

$$(9,9,9) \xrightarrow{(s/s)} (0,0,0)$$

occurs in one step.

Notice that the previous example shows explicitly the compositionality of our algebraic model, and it does not seem that this feature could be exploited to represent the Decimal Counter in such a natural way as in $\mathbf{Span}(\mathbf{Graph})$ by using other formalisms known in the literature.

4. A biological application: the heart

In this section we provide a simplified model of a human heart [9]. The heart is a muscular organ and it is the engine of the blood flow. Being a physical system, it could be modeled "globally" using continuous time and differential equations, but, we believe, with great difficulty. In our approach we model it *compositionally* as a discrete dynamical system. We observe that we focus on the regular behavior of the heart. Atypical behavior (for example cardiac arrhythmia) could be described in a similar way. Considering that the heart system is quite complex, we concentrate first on formalizing its basic components: the heart's interfaces —veins, aorta and pulmonary arteries—, atria, ventricula, tricuspid/mitral valve, sinoatrial node, atrioventricular node and the His bundle.

The heart comprehends arterial blood vessels —aorta on the left and pulmonary artery on the right— and two veins. In the right side of the heart, blood continuously enters through the veins and comes out through the pulmonary artery, whereas on the left side enters in the pulmonary vein and comes out from the aorta. So the heart's interfaces are:

- Pulmonary vein on the left side of the heart;
- Aorta on the left side of the heart;
- Superior vena cava and Inferior vena cava on the right side of the heart;
- Pulmonary artery on the right side of the heart.

We emphasize that blood is *continuously* going through the heart system, so it would be incorrect to describe its flux in terms of input/output events.

The heart system furthermore consists of four cavity, paired up in two similar subsystems, left and right. Each subsystem includes an atrium, on the top, and a ventricle, on the bottom. Atria related to ventricles through tricuspid —for the right side— and mitral —for the left side— valve. Atria are like a tank of blood coming from veins. After a beat, the blood continues to enter, increasing the internal pressure until a threshold is reached, that causes the tricuspid/mitral valve to open. The blood begins to drain through the tricuspid/mitral valve and at the same time the internal pressure in the atria decreases. There is a further pressure peak in the atria due to the electric signal from the sinoatrial node which causes the atrium's contraction and the blood's flow through the tricuspid/mitral valve.

The sinoatrial node generates the normal rhythmic impulse and distributes this impulse to both the atria, in a simultaneous way. The normal range is 60, 80 beats per minute. The atrioventricular node is primarily responsible for the delay in passing the signal from the atria to the ventricles, whereas the His bundle propagates the impulse to the ventricular heart mass.

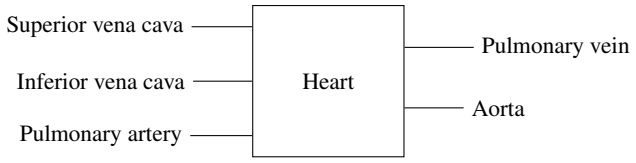


Figure 1. Heart

It should be noted that the operation of the heart, shown below, is a first approximation that can be extended by considering for example other components, like semilunar valves —they are located in the tricuspid/mitral valve and prevent reflux—, time and probability. First, we give a high level view of the heart in Figure 1. Next, we give more detail concerning the heart in Figure 2.

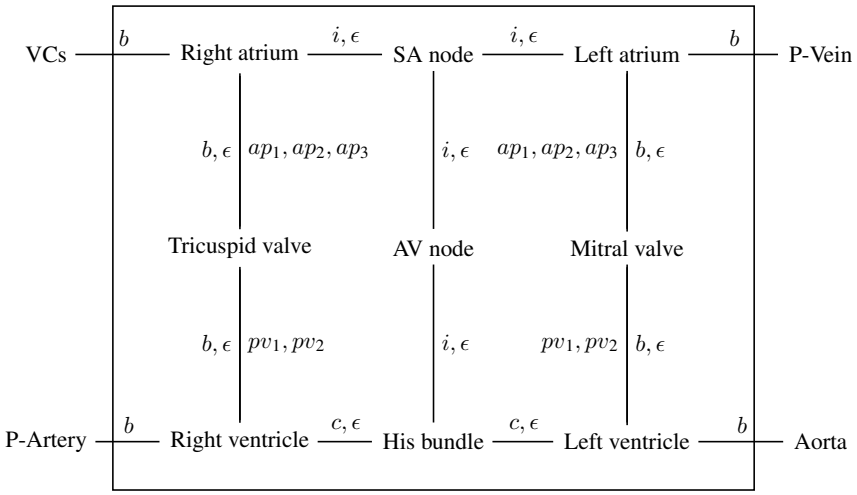


Figure 2. Architecture

Figure 3 shows the functioning of the heart's interfaces —in our simplified example— which are very simple automata with alphabet $\{b\}$. Veins, aorta and pulmonary artery have one state and one transition, labeled with b. The idea is that a heart's interface —veins, aorta or pulmonary artery— is continuously broadcasting blood b in a way which does not impede blood's flow to and from the heart.

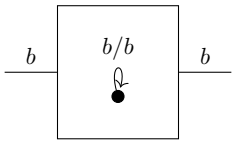


Figure 3. Veins, aorta or pulmonary artery

In the following, we adopt the convention to denote interfaces (by abuse of notation) with *Component*=$\{$*labels*$\}$, where *labels* are the proper labels of the interfaces and *Component* corresponds to the

automaton to be connected. Now we describe (an abstraction of) the two atria. For the sake of simplicity, we present only the right part —Figure 4— : the other side and also the ventricles' modeling are straightforward and then left to the reader. It has one interface named and labeled by the cartesian product SVC $\times$ IVC = $\{b\}$ ("b" stands for "blood") —Superior vena cava $\times$ Inferior vena cava—, another interface named and labeled by the Sinoatrial (SA) node= $\{i,\epsilon\}$ ("i" stands for "impulse"), and two interfaces (linked to the tricuspid valve) $\{ap_1, ap_2, ap_3\}$ (atrial pressure) and $\{b, \epsilon\}$ (blood). The right atrial has states AP_1, AP_2, AP_3 which represent the atrial pressure: AP_1 represents the lowest pressure (when the tricuspid valve is closed), AP_2 relates to the blood flow inside the tricuspid valve and AP_3 is the highest state (what you get when the SA node sends an impulse to the right atrium). The transitions are:

b,ϵ/ap$_1$,ϵ : AP$_1$$\rightarrow$AP$_1$, b,$\epsilon$/ap$_2$,$\epsilon$: AP$_1$$\rightarrow$AP$_2$,
b,ϵ/ap$_2$,b : AP$_2$$\rightarrow$AP$_2$, b,i/ap$_3$,b : AP$_2$$\rightarrow$AP$_3$,
b,ϵ/ap$_2$,b : AP$_3$$\rightarrow$AP$_2$, b,$\epsilon$/ap$_1$,$\epsilon$: AP$_2$$\rightarrow$AP$_1$

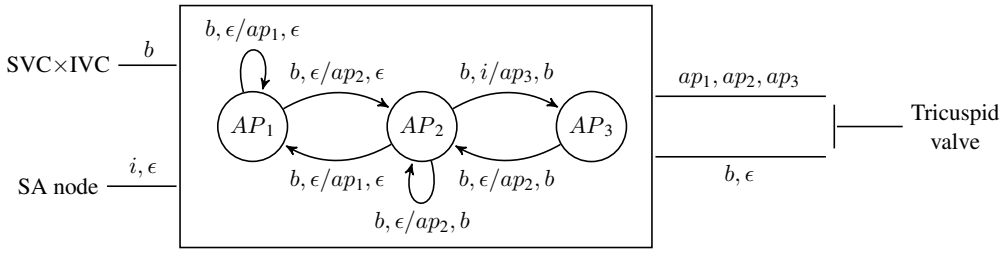


Figure 4. Right atrium (R-Atrium)

Next, we describe the tricuspid/mitral valve, in Figure 5. The interfaces are named and labeled by Atrium$_1$=$\{ap_1, ap_2, ap_3\}$ (atrial pressure), Atrium$_2$=$\{b, \epsilon\}$ (blood), Ventricle$_1$=$\{vp_1, vp_2\}$ (ventricular pressure), Ventricle$_2$=$\{b, \epsilon\}$ (blood). The states are $Open, Close, Peak$, and the transitions are:

$ap_1, \epsilon/pv_1, \epsilon : Close \rightarrow Close$, $ap_2, \epsilon/pv_1, \epsilon : Close \rightarrow Open$,
$ap_2, b/pv_1, b : Open \rightarrow Open$, $ap_3, b/pv_1, b : Open \rightarrow Peak$,
$ap_3, b/pv_1, b : Peak \rightarrow Peak$, $ap_1, \epsilon/pv_2, \epsilon : Peak \rightarrow Close$

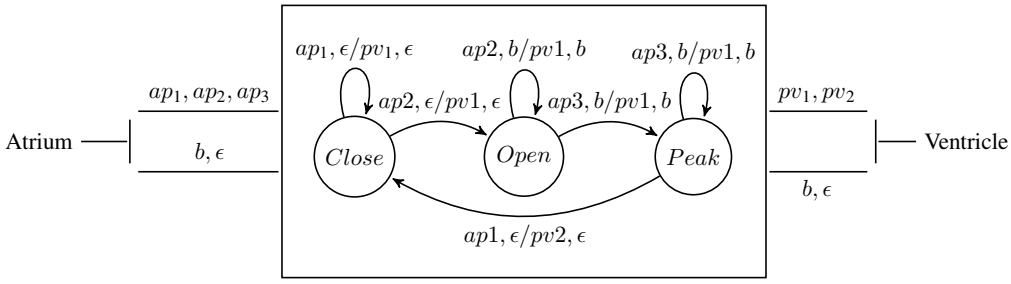


Figure 5. Tricuspid valve (Tri-Valve)

The Sinoatrial node (SA node) regulates the heart rate through a signal that sends an impulse to the right atrium, the left atrium and the AV node. The normal rate is 60 or 80 beats per minute. We start with 60 bpm (see Figure 6), which is one beat per second. We want to temporize the discrete model used in our formalism, so we assume that the impulse *i* occurs with an atomic action that lasts for one second. Alternatively, we model it with an non-atomic action consisting of [start(impulse), waiting, end(action)], always of one second. It may be that the SA node transmits that impulse, impulse of contractions, at a higher or lower frequency. By way of example, in Figure 7, we have a SA node at a frequency of 60 bpm (so the SA node transmits *i* at a frequency of 1 bps) or 80 bpm (which is 0.75 bps). The SA node has only three interfaces: right atrium, left atrium and the AV node. In Figure 6 there are $\{i, \epsilon\}$ (impulse) and there are 60 states and transitions: one transition is $/i, i, i : 1 \to 2$ and other transitions are $/\epsilon, \epsilon, \epsilon : s \to s'$. On the other hand, in Figure 7, there are 139 (60+80-1) states which refer to the two different heart rates (60 or 80 bpm), the interfaces are $\{i_1, i_2, \epsilon\}$ and its function is the same as Figure 6. It can be seen that the SA node component could be modeled in a more elegant way using CospanSpan(Graph), as in the example of Sofia's Birthday Party [7, 11]. Intuitively, when a certain state is reached, the SA node component can activate a similar component working in a different modality, i.e. with different beat rating.

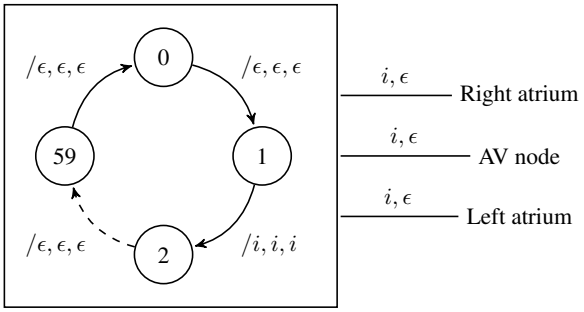


Figure 6. SA node (60 bpm)

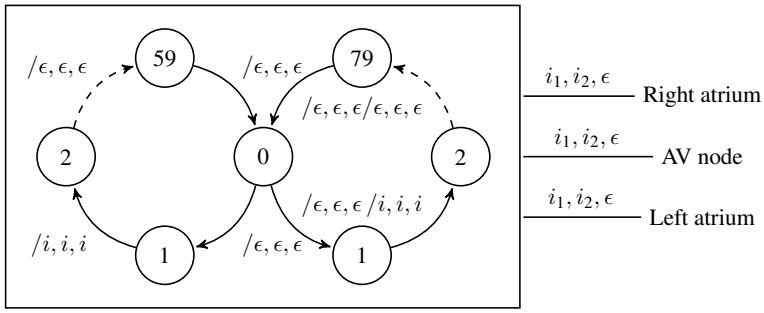


Figure 7. SA node (60/80 bpm)

Now we describe the Atrioventricular node (Figure 8). The AV node sees the impulse generated by the SA node and retransmits it, after a delay, to the His bundle. The AV node sends the pulses at different frequencies, which we can indicate with $i_1, i_2, \ldots$ It has one interface named and labeled by the SA node = $\{i, \epsilon\}$ and one interface named and labeled by the His bundle = $\{i, \epsilon\}$. The transitions are:

$\epsilon/\epsilon : 0 \to 0$, $i/\epsilon : 0 \to 1$,
$\epsilon/\epsilon : 1 \to 2$, $\epsilon/\epsilon : 2 \to 3$,
... ...
$\epsilon/\epsilon : (n-1) \to n$, $\epsilon/i : n \to 0$

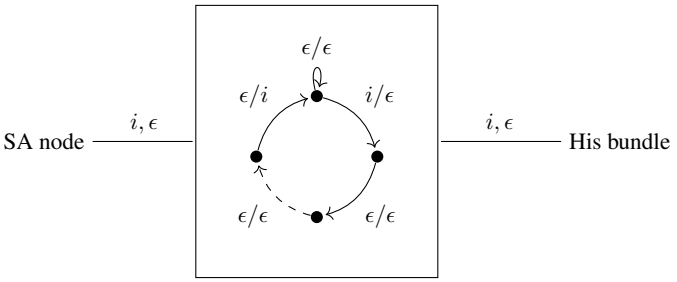


Figure 8. AV node

Finally, we present the His bundle, in Figure 9. It has one interface named and labeled by the AV node = $\{i, \epsilon\}$, and two other interfaces named and labeled by the right ventricle = $\{c, \epsilon\}$ and the left ventricle = $\{c, \epsilon\}$. It has one state and two different transitions: $i/c, c : s \to s$ and $\epsilon/\epsilon, \epsilon : s \to s$.

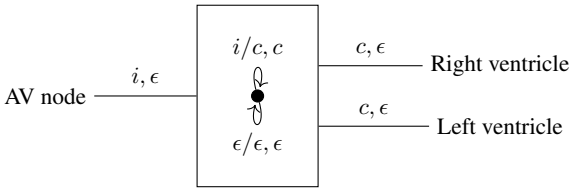


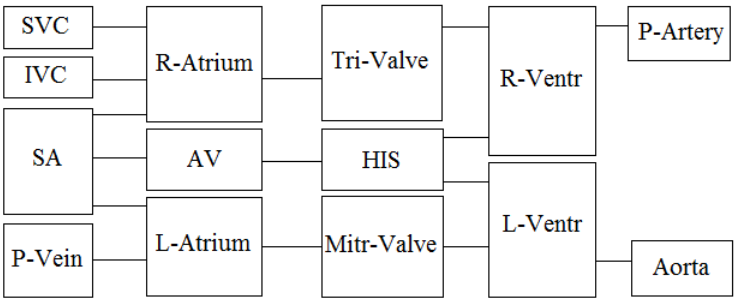
Figure 9. His bundle (HIS)

We give the algebraic representation of the whole automaton of the heart:

$$(\text{SVC} \otimes \text{IVC} \otimes \text{SA} \otimes \text{P-Vein}) \cdot (\text{R-Atrium} \otimes \text{AV} \otimes \text{L-Atrium}) \cdot$$

$$\cdot(\text{Tri-Valve} \otimes \text{HIS} \otimes \text{Mitr-Valve}) \cdot (\text{R-Ventr} \otimes \text{L-Ventr}) \cdot (\text{P-Artery} \otimes \text{Aorta})$$

This algebraic expression corresponds to the following geometric representation of the whole automaton (as explained in Section 2.1):



Notice that the previous picture is basically the transpose of the one in Figure 2 since we assumed to always represent restricted products horizontally and tensor products vertically.

5. Conclusions and future work

In this paper, we give a survey of the algebra of CospanSpan(Graph), focusing on the compositional feature of this algebraic formalism. In several previous works, we showed that this algebra provides a natural description for different kinds of discrete dynamical systems, both artificial (e.g., train control, distributed sort algorithm etc.) and physical (e.g., Decimal Counter). As a novel application, we consider a biological system: a compositional description of the heart in Span(Graph).

The model that we presented describes the "correct" behavior of the heart in a healthy human being. Using Cospan(Graph), it is possible to model alterations in the normal behavior, like cardiac arrhythmia. In fact, the sequential composition in Cospan(Graph) allows us to compose different automata, with the same parallel interfaces, modeling a variation of a part of the system over the time. Indeed, the role of Cospan(Graph) in the general algebra is similar to the Kleene's algebra for classical sequential automata. Hence, the variation between the normal behavior and the pathological one could be easily formalized in CospanSpan(Graph).

Since our model is compositional, we leave for future work a complete description of the whole heart-pacemaker system that can be provided in a timed extension of CospanSpan(Graph).

References

[1] Besozzi D, Mauri G, Păun G, Zandron C. Gemmating P systems: collapsing hierarchies. *Theor. Comput. Sci.*, 2003. **296**(2):253–267. URL `https://doi.org/10.1016/S0304-3975(02)00657-6`.

[2] Martín-Vide C, GMauri, Paun G, Rozenberg G, Salomaa A (eds.). Membrane Computing, International Workshop, WMC 2003, Tarragona, Spain, July 17-22, 2003, Revised Papers, volume 2933 of *Lecture Notes in Computer Science*. Springer, 2004. ISBN-978-3-540-20895-2, 0302-9743. doi:10.1007/b95207.

[3] Ferretti C, Mauri G, Zandron C (eds.). DNA Computing, 10th International Workshop on DNA Computing, DNA 10, Milan, Italy, June 7-10, 2004, Revised Selected Papers, volume 3384. Springer, 2005. doi:10.1007/b136914.

[4] Katis P, Sabadini N, Walters RFC. Span(Graph): A Categorial Algebra of Transition Systems. In: Proc. of AMAST. 1997 pp. 307–321. doi:10.1007/BFb0000479.

[5] Katis P, Sabadini N, Walters R. A Formalization of the IWIM Model. In: Proc. of Coordination Languages and Models, volume 1906 of *LNCS*. Springer. 2000 pp. 267–283. ISBN-978-3-540-41020-1.

[6] Sabadini N, Walters RFC. Hierarchical Automata and P-systems. *Electr. Notes Theor. Comput. Sci.*, 2003. **78**:5–19. URL `https://doi.org/10.1016/S1571-0661(04)81003-X`.

[7] de Francesco Albasini L, Sabadini N, Walters RFC. The Compositional Construction of Markov Processes. *Applied Categorical Structures*, 2011. **19**(1):425–437. doi:10.1007/s10485-010-9233-0.

[8] Cherubini A, Sabadini N, Walters RFC. Timing in the Cospan-Span Model. *Electr. Notes Theor. Comput. Sci.*, 2004. **104**:81–97. doi:10.1016/j.entcs.2004.08.020.

[9] Jiang Z, Pajic M, Moarref S, Alur R, Mangharam R. Modeling and Verification of a Dual Chamber Implantable Pacemaker. In: Proc. of TACAS. 2012 pp. 188–203. doi:10.1007/978-3-642-28756-5_14.

[10] Bènabou J. Introduction to Bicategories. *Reports of the Midwest Category Seminar*, 1967. **47**:1–77. doi:10.1007/BFb0074299.

[11] Gianola A, Kasangian S, Sabadini N. Cospan/Span(Graph): an Algebra for Open, Reconfigurable Automata Networks. In: Proc. of CALCO. 2017 pp. 2:1–2:17. doi:10.4230/LIPIcs.CALCO.2017.2.

[12] Sabadini N, Schiavio F, Walters R. On the geometry and algebra of networks with state. *Theor. Comput. Sci.*, 2017. **664**:144–163. URL `https://doi.org/10.1016/j.tcs.2016.01.028`.

[13] Zielonka W. Notes on Finite Asynchronous Automata. *ITA*, 1987. **21**(2):99–135. URL `https://doi.org/10.1051/ita/1987210200991`.