

Circuits of finite memory retrospective operators

A. P. Francisco

CLC/DEI, IST, Universidade Técnica de Lisboa

Av. Rovisco Pais, 1049-001 Lisbon, Portugal

`alexandre.francisco@dei.ist.utl.pt`

June 11, 2003

Abstract

In this work we introduce a theory of circuits of finite memory retrospective operators to fully characterize the class of finite memory retrospective operators, expanding the theory developed by B. A. Trakhtenbrot and A. Rabinovich about automata over continuous time. A physical device, in which complex transformations are implemented, is usually an appropriate combination of elementary parts that interact as desired. These ideas conduct us to the concept of circuit which appears many times in literature. Our main contribution to this theory is the introduction of circuits of finite memory retrospective operators over signals, i.e., we choose a set of elementary finite memory retrospective operators and we study how to obtain all finite memory retrospective operators by constructing circuits with the elementary operators.

1 Introduction

Nowadays control systems are commonly found in many devices such as industrial robots and airplanes. The study of these devices and their interaction with their environment leads us to the mathematical theory of control, which deals with the analysis and design of control systems.

As we can see in [Son90], there have been two main lines of work in control theory. One take us over the models in mathematics and physics and over their possible optimization with respect to a particular behavior. The other main line is based on the constraints of a specific object and the goal is to correct the deviations of such object from a desired behavior.

So and specially in what concerns the second main line, we need to study systems which involve interacting networks of digital and continuous systems, i.e., hybrid systems. These incorporate both discrete and continuous dynamics in which the continuous aspects may require incursions into calculus and differential equations. As we know, differential equations have nothing in common with existing and well understood tools of automata theory and logic.

The approximation between automata theory and continuous systems lead us to extensions of the basic finite automata paradigm. A first extension arise from the idea of interaction with environment seen as an oracle. A second one focuses on the use of continuous time instead of discrete time and ignores interaction with the oracle. As is put in [RT98], it is believed that these two orthogonal

extensions may facilitate a structured formalization of hybrid systems and a lucid adaptation of basic automata theory to hybrid systems.

In this work we will be around the second extension and we will follow ideas discussed by B. A. Trakhtenbrot and A. Rabinovich, see e.g. [Rab97, RT98, Tra98, Tra99]. Their work include many definitions and formalizations with respect to automata over continuous time, namely with respect to lift concepts of the classical automata theory from discrete to continuous time. We note also that this work is specially related to speed independent properties which rely on the order of real numbers, metric aspects which deal with the distance between real numbers are not considered.

R. Alur and D. L. Dill (see, e.g., [AD94]) have proposed timed automata to model the behavior of real-time systems and in their approach metric properties of the reals are taken into account. In fact the work of R. Alur and D. L. Dill deals with timed automata, finite automata which have a finite number of clocks associated, that accept words in which real-valued time of occurrence is associated with each symbol. I.e., a timed automaton accepts words as a classical automaton with a major difference: the time between the occurrence of two symbols is not necessarily constant and may be a real number. Note that, in the approach taken by B. A. Trakhtenbrot and A. Rabinovich, signals over real-time are considered in place of words and automata compute operators on signals in place of words accepted.

A physical device, in which complex transformations are implemented, is usually an appropriate combination of elementary parts that interact as desired. This idea guides us to the concept of circuit which appears many times in literature (see, e.g., [KT65]) and which permits, for example, to describe a given automaton as the combination of elementary automata.

Our main contribution to this theory is the introduction of circuits of finite memory retrospective operators over signals, i.e., we choose a set of elementary finite memory retrospective operators and we study how to obtain all finite memory retrospective operators by constructing circuits with the elementary operators. In order to perform this study, we use the notion of function algebra in [Clo99] and we obtain an algebra of finite memory retrospective operators. The equivalence between this algebra of operators and the set of finite memory retrospective operators is stated in propositions 5.1 and 4.15. We note that the proof of proposition 5.1 is constructive and so, making use of the elementary operators and operations provided by this algebra, we can construct circuits of operators.

So, in the first section we will introduce some definitions and concepts about automata over continuous time that were previously introduced in [Tra98] and [Rab97]. In section three we will specify the set \mathcal{X} of primitive operators and verify that in fact \mathcal{X} is a set of finite memory retrospective operators. The set of operations OP will be given in the fourth section and some properties involving them will be stated. The main result will arise in the fifth section, namely will be proved that the function algebra specified, $[\mathcal{X}, OP]$, coincide precisely with the class of finite memory retrospective operators, \mathcal{O}_{FMR} . In the sixth section we provide examples of circuits for some finite memory retrospective operators. The correspondence between circuits and automata is stated in the last section.

2 Preliminaries

A function $x : \mathbb{R}^+ \rightarrow \Sigma$ is a signal over Σ , where Σ is a finite set. A non-Zeno signal x is a piecewise constant function $x : \mathbb{R}^+ \rightarrow \Sigma$ and we denote the set of non-Zeno signals over an alphabet Σ by $nZSig(\Sigma)$.

A function from signals to signals is called a signal operator. In [Rab97, PRT01] some notions related to operators on signals have been introduced.

Definition 2.1 (Speed Independence) An operator F from signals to signals is speed independent if for every order preserving bijection ρ on time:

$$\forall \vec{x}, F((x_1 \circ \rho), \dots, (x_n \circ \rho)) = F(\vec{x}) \circ \rho.$$

Definition 2.2 ((Strong) Retrospective Operator)

Let $F : nZSig(\Sigma)^n \rightarrow nZSig(\Sigma)$ be an operator from signals to signals. F is retrospective with respect to the i -th argument if for any $\vec{x}, \vec{y} \in nZSig(\Sigma)^n$ such that $x_j = y_j$, for $i \neq j$, and $t \in \mathbb{R}^+$ the following condition holds: if x_i and y_i coincide in the interval $[0, t]$, then $F\vec{x}$ and $F\vec{y}$ coincide in the interval $[0, t]$. F is strong retrospective with respect to the i -th argument if for any $\vec{x}, \vec{y} \in nZSig(\Sigma)^n$ such that $x_j = y_j$, for $i \neq j$, and $t \in \mathbb{R}^+$ the following condition holds: if x_i and y_i coincide in the interval $[0, t]$, then $F\vec{x}$ and $F\vec{y}$ coincide in the interval $[0, t]$. Given a set $S \subseteq \{1, \dots, n\}$ of components, an operator F is (strong) retrospective with respect to S when it is (strong) retrospective with respect to i for all $i \in S$. An operator F is (strong) retrospective when it verifies the above conditions for all components i .

Notation 2.3 Let $x, z \in nZSig(\Sigma)$. The *concatenation* of the t -prefix of the signal x with the signal z (notation $x \upharpoonright^t; z$) is defined as:

$$(x \upharpoonright^t; z)(\tau) = \begin{cases} x(\tau) & \text{if } \tau < t \\ z(\tau - t) & \text{if } \tau \geq t \end{cases}$$

Definition 2.4 (Residual [Rab97]) Let $F : nZSig(\Sigma)^n \rightarrow nZSig(\Sigma)$ be an operator on signals, $\vec{x} \in nZSig(\Sigma)^n$ a signal, and $t \in \mathbb{R}^+$ a time point. The residual of F with respect to \vec{x} and t , $Res(F, \vec{x}, t)$, is the operator:

$$\lambda z_1 \dots z_n. \lambda t'. F((x_1 \upharpoonright^t; z_1), \dots, (x_n \upharpoonright^t; z_n))(t + t').$$

Definition 2.5 (Finite/Countable Memory [Rab97]) An operator F is a finite (countable) memory operator if it has finitely (countably) many distinct residuals, i.e., if the set $\{Res(F, \vec{x}, t) : \vec{x} \in nZSig(\Sigma)^n, t \in \mathbb{R}^+\}$ is finite (countable).

In [Rab97] were also introduced notions about the characterization of non-Zeno signals and about the characterization of functions over non-Zeno signals. A non-Zeno signal x over an alphabet Σ is said to be characterized by α, α', τ if $\alpha = \langle a_i : i \in \mathbb{N} \rangle$ and $\alpha' = \langle a'_i : i \in \mathbb{N} \rangle$ are ω -strings over Σ^n , $\tau = \langle t_i : i \in \mathbb{N} \rangle$ is a time scale¹ and $x(t_i) = a_i$ and $x(t) = a'_i$, for every $i \in \mathbb{N}$ and every $t \in]t_i, t_{i+1}[$. An operator F from non-Zeno signals over Σ to non-Zeno signals over Σ is said to be characterized by a function $G : (\Sigma^n)^\omega \times (\Sigma^n)^\omega \rightarrow \Sigma^\omega \times \Sigma^\omega$ if whenever α, α', τ characterizes \vec{x} then $G(\alpha, \alpha'), \tau$ characterize $F\vec{x}$.

¹An unbounded increasing sequence $t_1 < t_2 < \dots$ such that $t_i \in \mathbb{R}^+$ and $t_1 = 0$ is called a time scale.

Definition 2.6 (Generalized SI Condition [Rab97]) Let G be a function on $(\Sigma^n)^\omega \times (\Sigma^n)^\omega$, let α and α' be ω -strings and let $i \in \mathbb{N}$. Let β and β' be obtained from α and α' by inserting a'_i immediately after the i -th position. Similarly, let ζ and ζ' be obtained from ξ and ξ' by inserting x'_i immediately after the i -th position. Then:

$$G(\alpha, \alpha') = (\xi, \xi') \text{ if and only if } G(\beta, \beta') = (\zeta, \zeta').$$

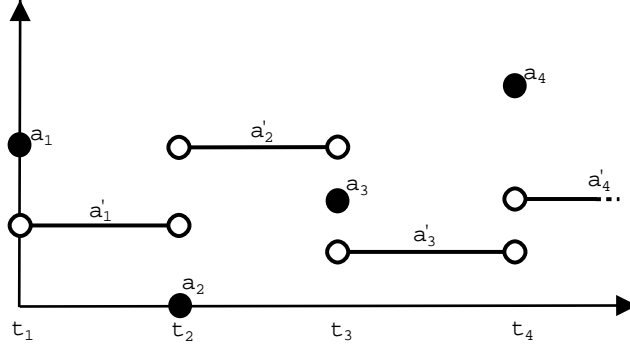


Figure 1: A non-Zeno signal.

Definition 2.7 (Generalized Retrospective Condition [Rab97]) Let G be a function on $(\Sigma^n)^\omega \times (\Sigma^n)^\omega$, and let $\alpha = \langle a_i : i \in \mathbb{N} \rangle$, $\alpha' = \langle a'_i : i \in \mathbb{N} \rangle$, $\beta = \langle b_i : i \in \mathbb{N} \rangle$, $\beta' = \langle b'_i : i \in \mathbb{N} \rangle$, $\xi = \langle x_i : i \in \mathbb{N} \rangle$, $\xi' = \langle x'_i : i \in \mathbb{N} \rangle$, $\zeta = \langle z_i : i \in \mathbb{N} \rangle$ and $\zeta' = \langle z'_i : i \in \mathbb{N} \rangle$ be ω -strings. Then G satisfies the *generalized retrospective condition* if

1. G is retrospective;
2. if $G(\alpha, \alpha') = (\xi, \xi')$, $G(\beta, \beta') = (\zeta, \zeta')$, $a_j = b_j$, for $j \in \{0, \dots, i\}$, and $a'_j = b'_j$, for $j \in \{0, \dots, i-1\}$, then $x_j = z_j$, for $j \in \{0, \dots, i\}$, and $x'_j = z'_j$, for $j \in \{0, \dots, i-1\}$.

The following proposition was stated in [Rab97].

Proposition 2.8 ([Rab97])

1. Every speed independent operator F on non-Zeno signals over Σ is characterized by a function G on $(\Sigma^n)^\omega \times (\Sigma^n)^\omega$ that satisfies the generalized SI condition.
2. Every function G on $(\Sigma^n)^\omega \times (\Sigma^n)^\omega$ that satisfies the generalized SI condition characterizes a speed independent operator F over non-Zeno signals.
3. If G is a function on $(\Sigma^n)^\omega \times (\Sigma^n)^\omega$ that characterizes a speed independent operator over non-Zeno signals F , then:
 - (a) F is retrospective if and only if G satisfies the generalized retrospective condition;
 - (b) F has finite memory if and only if G has finite memory;
 - (c) F has countable memory if and only if G has countable memory.

3 Primitive Operators

Intuitively we may consider some operators that seem to be most relevant or basic. Given a finite memory retrospective operator F over non-Zeno signals, its value at an instant t depends on the past values of the signal \vec{x} given as argument. Thus, in order to compute the value of F at t , it will be useful to know the behavior of \vec{x} before t , namely the last jump² of \vec{x} before t and the left limit³ of \vec{x} at t .

Following these intuitions, we define now two primitive operators which are needed to get the main characterization results.

Definition 3.1 (Left Limit) Let Σ be a finite set and $a \in \Sigma$. We define the unary operator $LLim_a : nZSig(\Sigma) \rightarrow nZSig(\Sigma)$ as follows:

$$LLim_a(x)(t) = \begin{cases} a & \text{if } t = 0 \\ b & \text{if } t > 0 \text{ and } b \text{ is the left limit of } x \text{ at } t. \end{cases}$$

Definition 3.2 (Last Jump Value) Let Σ be a finite set, $a \in \Sigma$, and $n, k \in \mathbb{N}$ such that $k \leq n$. We define the n -ary operator $LJV_{k,a}^n : nZSig(\Sigma)^n \rightarrow nZSig(\Sigma)$ as follows:

$$LJV_{k,a}^n(\vec{x})(t) = \begin{cases} a & \text{if } t = 0 \\ x_k(\tau) & \text{if } t > 0 \text{ and } \tau < t \text{ are such that } \vec{x} \text{ is constant} \\ & \text{in }]\tau, t[\text{ and } \vec{x} \text{ is not continuous at } \tau. \end{cases}$$

Since the left limit is undefined at the instant 0 and there is no jumps before that, we have introduced default values at $t = 0$ in the above definitions. Therefore, we grant that $LLim_a$ and $LJV_{k,a}^n$ are well defined at $t = 0$.

These operators are evidently strong retrospective operators. We shall see that they are finite memory operators.

Proposition 3.3 *$LLim_a$ is a finite memory operator.*

Proof: Let $x \in nZSig(\Sigma)$ and $t \in \mathbb{R}^+$, the residual G of $LLim_a$ with respect to x and t is defined as follows:

$$\begin{aligned} G(z)(t') &= LLim_a(x]t; z)(t + t') \\ &= LLim_b(z)(t') \end{aligned}$$

where b is the left limit of x at t whenever $t > 0$, otherwise $b = a$. Thus, the set of residuals of $LLim_a$ is given by $\{LLim_a : a \in \Sigma\}$. Since Σ is a finite set, the set of residuals is finite and $LLim_a$ is a finite memory operator. ■

Proposition 3.4 *$LJV_{k,a}^n$ is a finite memory operator.*

²Let \vec{x} be a signal, the last jump of \vec{x} before $t \in \mathbb{R}^+$ is c if there exist $t', t'' \in \mathbb{R}^+$ such that $t'' < t' < t$, $x(t') = c$, $x(\tau) = c'$ for $\tau \in]t', t[$, $x(\tau) = c''$ for $\tau \in]t'', t'[$ and $c \neq c'$ or $c \neq c''$.

³Let x be a signal, x has left limit c at $t \in \mathbb{R}^+$ if there exists $t' \in \mathbb{R}^+$ such that $t' < t$ and $x(\tau) = c$ for $\tau \in]t', t[$.

Proof: Let $\vec{x} \in nZSig(\Sigma)^n$ and $t \in \mathbb{R}^+$. We know that the residual G of $LJV_{k,a}^n$ with respect to \vec{x} and t is given by

$$\begin{aligned} G(\vec{z})(t') &= LJV_{k,a}^n((x_1]_t^t; z_1), \dots, (x_n]_t^t; z_n)(t + t') \\ &= F_{k,b,\xi}^n(\vec{z})(t') \end{aligned}$$

where $b = LJV_{k,a}^n(\vec{x})(t)$, $\xi = \langle c_1, \dots, c_n \rangle = \langle LLim_a(x_i)(t) : i = 1, \dots, n \rangle$, and $F_{k,b,\xi}^n$ is defined as follows:

$$F_{k,b,\xi}^n(\vec{z})(t') = LJV_{k,b}^n((Jump_{b \rightarrow c_1}]_1^1; z_1), \dots, (Jump_{b \rightarrow c_n}]_1^1; z_n)(t' + 1).$$

Therefore the set of residuals of $LJV_{k,a}^n$ is given by $\{F_{k,b,\xi}^n : b \in \Sigma \text{ and } \xi \in \Sigma^n\}$. Since Σ is a finite set, the set of residuals is finite, and $LJV_{k,a}^n$ is a finite memory operator. ■

Definition 3.5 (Pointwise Extension) If $f : \Sigma^n \rightarrow \Sigma'$ is a total function where Σ and Σ' are both finite non-empty sets, then we define the operator $P_f : nZSig(\Sigma)^n \rightarrow nZSig(\Sigma')$ as the pointwise extension of f as follows:

$$P_f(\vec{x})(t) = f(\vec{x}(t))$$

Since the value of these operators only depends on the current instant, we see that these operators are retrospective and have finite memory.

Definition 3.6 (Set of Primitives) Given $\Sigma_1, \dots, \Sigma_m$ finite non-empty sets, the set $\mathcal{X}[\Sigma_1, \dots, \Sigma_m]$ of primitives is defined as follows:

$$\begin{aligned} \mathcal{X}[\Sigma_1, \dots, \Sigma_m] &= \{LLim_a : a \in \Sigma_i, i = 1, \dots, m\} \\ &\cup \{LJV_{k,a}^n : a \in \Sigma_i, i = 1, \dots, m \text{ and } k \leq n\} \\ &\cup \{P_f : n \in \mathbb{N}, f : \Sigma_i^n \rightarrow \Sigma'_j, i, j = 1, \dots, m\} \end{aligned}$$

4 Operations

We know that the set of finite memory retrospective operators is closed under composition and naturally we will consider the composition \circ as one of the available operations.

Definition 4.1 (Composition) Let $G : nZSig(\Sigma')^m \rightarrow nZSig(\Sigma'')$ be a m -ary finite memory retrospective operator and let $F_i : nZSig(\Sigma)^n \rightarrow nZSig(\Sigma')$ be also a n -ary finite memory operator, for $i = 1, \dots, m$. We define the composition of G with F_1, \dots, F_m , $\circ(G, F_1, \dots, F_m) : nZSig(\Sigma)^n \rightarrow nZSig(\Sigma'')$, as follows:

$$\circ(G, F_1, \dots, F_m)(\vec{x}) = G(F_1(\vec{x}), \dots, F_m(\vec{x})).$$

Clearly, sometimes we need to know previous values of an operator to get its current value. Thus, we define next the operation *Rec*.

Definition 4.2 (Rec) Let $G : nZSig(\Sigma)^{n+1} \rightarrow nZSig(\Sigma)$ be a finite memory retrospective operator that verify the following conditions:

- is strong retrospective with respect to the last argument;
- given $x_1, \dots, x_n, y, z \in nZSig(\Sigma)$, $t > 0$, and $t' \in [0, t[$ such that x_i is constant in $]t', t[$, for all i , and x_j is not continuous at t' for some j , if $y(\tau) = z(\tau)$, for $\tau \in [0, t'[$, then $G(x_1, \dots, x_n, y)(t) = G(x_1, \dots, x_n, z)(t)$.

For an arbitrary⁴ $a \in \Sigma$, the operation Rec is defined as follows:

$$Rec(G)\vec{x}t = \begin{cases} G(x_1, \dots, x_n, Const_a)t & \text{if } t = 0 \\ G(x_1, \dots, x_n, (Rec(G)(\vec{x})]^{t'}; Const_a))t & \text{if } t > 0 \end{cases}$$

and clearly $Rec(G) : nZSig(\Sigma)^n \longrightarrow nZSig(\Sigma)$.

We will prove that the operators obtained with Rec , are well defined and are finite memory retrospective operators.

Given the operator $H_{\vec{x}} : nZSig(\Sigma) \longrightarrow nZSig(\Sigma)$ such that $H_{\vec{x}}(y) = G(\vec{x}, y)$, for $\vec{x} \in nZSig(\Sigma)$, we will study the fixpoints of $H_{\vec{x}}$ as fixpoints of a function over ω -strings. We observe that $Rec(G)(\vec{x})$ is a fixpoint of $H_{\vec{x}}$.

Proposition 4.3 *Let G be in the conditions of definition 4.2 and let $\vec{x} \in nZSig(\Sigma)^n$. The operator $H_{\vec{x}}$, such that $H_{\vec{x}}(y) = G(\vec{x}, y)$, for $y \in nZSig(\Sigma)$, is characterized by a function $K_{\vec{x}} : \Sigma^\omega \times \Sigma^\omega \longrightarrow \Sigma^\omega \times \Sigma^\omega$ with the property:*

if $\alpha_i = \beta_i$ and $\alpha'_i = \beta'_i$, for $i < n$, then $K(\alpha, \alpha')_i = K(\beta, \beta')_i$, for $i \leq n$.

Proof: If G is a finite memory operator, then G is speed independent⁵. Thus $H_{\vec{x}}$ is a finite memory operator and by proposition 2.8 there exists a function $K_{\vec{x}} : \Sigma^\omega \times \Sigma^\omega \longrightarrow \Sigma^\omega \times \Sigma^\omega$ that characterizes $H_{\vec{x}}$.

In order to prove the above property let $\alpha, \alpha', \beta, \beta' \in \Sigma^\omega$ such that $\alpha_i = \beta_i$ and $\alpha'_i = \beta'_i$, for $i < n$. Suppose also that y is the non-Zeno signal characterized by α, α', τ and z is the non-Zeno signal characterized by β, β', τ where the time scale $\tau = \langle t_i : i \in \mathbb{N}_0 \rangle$ satisfies:

if $t \leq t'$, then $t \in \tau$ iff $\exists i$, x_i is not constant at t ,

for $t' \in \mathbb{R}^+$ such that \vec{x} is not constant at t' and such that \vec{x} is constant at τ for $\tau > t'$ ⁶. Thus we verify that $y(t) = z(t)$, for $t \in [0, t_{n-1}[$, and then we conclude by the given properties of G in definition 4.2 that $H_{\vec{x}}(y)(t) = H_{\vec{x}}(z)(t)$ with $t \in [0, t_n[$. Therefore $K_{\vec{x}}(\alpha, \alpha')_i = K_{\vec{x}}(\beta, \beta')_i$, for $i \leq n$, because $K_{\vec{x}}$ is a characterization of $H_{\vec{x}}(y)$. ■

In the following results, some knowledge about partial orders, complete lattices and continuous operators is needed; we will assume that the reader have that knowledge. We need also work with prefixes, thus we recall the concept of prefix.

Definition 4.4 Let $\alpha \in \Sigma^\omega$ be a ω -string, we define the set of its prefixes as:

$$P_\alpha = \{u \in \Sigma^* : \exists \beta \in \Sigma^\omega, \alpha = u\beta\}$$

⁴By the given conditions, we may choose any a .

⁵This fact is stated and proved in [Rab97].

⁶Clearly, such t' may not exist. This condition express the minimality of the time scale τ needed to characterize the non-Zeno signal \vec{x} .

Definition 4.5 (Prefix Closed Set) A set $X \subseteq \Sigma^*$ is a prefix closed set whenever it satisfies the following condition:

$$\forall u \in X, \forall v \in \Sigma^*, \text{ if } u = sv, \text{ then } s \in X.$$

Let $X \subseteq \Sigma^*$ be a prefix closed set such that there exists $u \in X$ with length n , for all $n \in \mathbb{N}$. Thus, it is clear that there exists at least one $\alpha \in \Sigma^\omega$ such that $P_\alpha \subseteq X$. Moreover, if for all $u \in X$ there exists only one $a \in \Sigma$ such that $ua \in X$, then $X = P_\alpha$ for some $\alpha \in \Sigma^\omega$ and we say that X characterizes univocally⁷ α .

Proposition 4.6 Let $\mathcal{P} = \{X \subseteq \Sigma^* : X \text{ is a prefix closed set}\}$, then $\langle \mathcal{P} \times \mathcal{P}, \leq \rangle$ is a complete lattice with \leq defined as follows:

$$(X, X') \leq (Y, Y') \text{ iff } X \subseteq Y \text{ and } X' \subseteq Y'.$$

Proof: Clearly $\langle \mathcal{P} \times \mathcal{P}, \leq \rangle$ is a partial order.

Let $\mathcal{S} \subseteq \mathcal{P} \times \mathcal{P}$ and $(Y, Y') \in \mathcal{P} \times \mathcal{P}$ such that, for all $(X, X') \in \mathcal{S}$, $(X, X') \leq (Y, Y')$. Then $\vee \mathcal{S} = (\bigcup_{(X, X') \in \mathcal{S}} X, \bigcup_{(X, X') \in \mathcal{S}} X') \leq (Y, Y')$, i.e., $\vee \mathcal{S}$ is the least upper bound of \mathcal{S} . Suppose now that $(Z, Z') \in \mathcal{P} \times \mathcal{P}$ is such that, for all $(X, X') \in \mathcal{S}$, $(Z, Z') \leq (X, X')$. Then $(Z, Z') \leq \wedge \mathcal{S} = (\bigcap_{(X, X') \in \mathcal{S}} X, \bigcap_{(X, X') \in \mathcal{S}} X')$, i.e., $\wedge \mathcal{S}$ is the greatest lower bound of \mathcal{S} . Clearly $\vee \mathcal{S} \in \mathcal{P} \times \mathcal{P}$ and $\wedge \mathcal{S} \in \mathcal{P} \times \mathcal{P}$, therefore $\langle \mathcal{P} \times \mathcal{P}, \leq \rangle$ is a complete lattice. ■

Definition 4.7 Let $K_{\bar{x}} : \Sigma^\omega \times \Sigma^\omega \longrightarrow \Sigma^\omega \times \Sigma^\omega$ be the function obtained in proposition 4.3. Given $a_1 a_2 \dots a_n, a'_1 a'_2 \dots a'_n \in \Sigma^*$, for $n \in \mathbb{N}$, we define the function $K'_{\bar{x}}$ as follows:

$$\begin{aligned} K'_{\bar{x}}(a_1 a_2 \dots a_n, a'_1 a'_2 \dots a'_n) &= (b_1 b_2 \dots b_{n+1}, b'_1 b'_2 \dots b'_{n+1}) \\ &\text{iff} \\ &\forall \alpha, \alpha' \in \Sigma^\omega, \exists \beta, \beta' \in \Sigma^\omega, \\ K_{\bar{x}}(a_1 a_2 \dots a_n \alpha, a'_1 a'_2 \dots a'_n \alpha') &= (b_1 b_2 \dots b_{n+1} \beta, b'_1 b'_2 \dots b'_{n+1} \beta'). \end{aligned}$$

Let $(X, X') \in \mathcal{P} \times \mathcal{P}$, we extend $K'_{\bar{x}}$ to $\mathcal{P} \times \mathcal{P}$ as follows:

$$K'_{\bar{x}}(X, X') = \{K'_{\bar{x}}(a_1 \dots a_n, a'_1 \dots a'_n) : a_1 \dots a_n \in X, a'_1 \dots a'_n \in X'\} \cup \{(\epsilon, \epsilon)\}.$$

Proposition 4.8 $K'_{\bar{x}}$ is well defined.

Proof: Let $a_1 a_2 \dots a_n, a'_1 a'_2 \dots a'_n \in \Sigma^*$, for $n \in \mathbb{N}$. Suppose that

$$K'_{\bar{x}}(a_1 a_2 \dots a_n, a'_1 a'_2 \dots a'_n) = (b_1 b_2 \dots b_{n+1}, b'_1 b'_2 \dots b'_{n+1}),$$

by definition 4.7, we have

$$\begin{aligned} &\forall \alpha, \alpha' \in \Sigma^\omega, \exists \beta, \beta' \in \Sigma^\omega, \\ K_{\bar{x}}(a_1 a_2 \dots a_n \alpha, a'_1 a'_2 \dots a'_n \alpha') &= (b_1 b_2 \dots b_{n+1} \beta, b'_1 b'_2 \dots b'_{n+1} \beta') \end{aligned}$$

and by proposition 4.3 we conclude that $(b_1 b_2 \dots b_{n+1}, b'_1 b'_2 \dots b'_{n+1})$ is univocally determined. Thus, the extension of $K'_{\bar{x}}$ to $\mathcal{P} \times \mathcal{P}$ is well defined. ■

⁷When we do not consider prefix closed sets, a ω -string α may be characterized by more sets than P_α , namely by any infinite subset of P_α .

Proposition 4.9 *Let $K'_{\vec{x}}$ be the extension obtained in definition 4.7. This function is defined from $\mathcal{P} \times \mathcal{P}$ to $\mathcal{P} \times \mathcal{P}$ and it is continuous.*

Proof: Let $(X, X') \in \mathcal{P} \times \mathcal{P}$. By the property of $K'_{\vec{x}}$ in proposition 4.3, we know that if $a_1 a_2 \dots a_n \in X$, $a'_1 a'_2 \dots a'_n \in X'$ and $K'_{\vec{x}}(a_1 a_2 \dots a_n, a'_1 a'_2 \dots a'_n) = (b_1 b_2 \dots b_n b_{n+1}, b'_1 b'_2 \dots b'_n b'_{n+1})$, then, for every $k \leq n$, $K'_{\vec{x}}(a_1 a_2 \dots a_k, a'_1 a'_2 \dots a'_k) = (b_1 b_2 \dots b_k b_{k+1}, b'_1 b'_2 \dots b'_k b'_{k+1})$. Thus, since $\{(\epsilon, \epsilon)\} \in K'_{\vec{x}}(X, X')$ by definition 4.7, $K'_{\vec{x}}(X, X')$ is closed under prefixes, i.e., $K'_{\vec{x}}(X, X') \in \mathcal{P} \times \mathcal{P}$.

Let $\mathcal{S} \subseteq \mathcal{P} \times \mathcal{P}$ be a directed subset. If $(X, X') \in \mathcal{S}$, then $K'_{\vec{x}}(X, X') \leq K'_{\vec{x}}(\vee \mathcal{S})$ where $\vee \mathcal{S} = (\bigcup_{(X, X') \in \mathcal{S}} X, \bigcup_{(X, X') \in \mathcal{S}} X')$. Clearly $\vee K'_{\vec{x}}(\mathcal{S}) \leq K'_{\vec{x}}(\vee \mathcal{S})$.

Suppose that $(b_1 b_2 \dots b_n b_{n+1}, b'_1 b'_2 \dots b'_n b'_{n+1}) \in \vee K'_{\vec{x}}(\mathcal{S})$, for some $n \in \mathbb{N}_0$, then there exists $(a_1 a_2 \dots a_n, a'_1 a'_2 \dots a'_n) \in \mathcal{S}$ such that $K'_{\vec{x}}(a_1 a_2 \dots a_n, a'_1 a'_2 \dots a'_n) = (b_1 b_2 \dots b_n b_{n+1}, b'_1 b'_2 \dots b'_n b'_{n+1})$. Thus, there exist $(X_1, X'_1), (X_2, X'_2) \in \mathcal{S}$ such that $a_1 a_2 \dots a_n \in X_1$ and $a'_1 a'_2 \dots a'_n \in X_2$. Because \mathcal{S} is directed, we know that there exists $(Y, Y') \in \mathcal{S}$ such that $(X_1, X'_1) \leq (Y, Y')$ and $(X_2, X'_2) \leq (Y, Y')$ and clearly $(b_1 b_2 \dots b_n b_{n+1}, b'_1 b'_2 \dots b'_n b'_{n+1}) \in K'_{\vec{x}}(Y, Y')$. As $(Y, Y') \leq \vee \mathcal{S}$, we have that $(b_1 b_2 \dots b_n b_{n+1}, b'_1 b'_2 \dots b'_n b'_{n+1}) \in K'_{\vec{x}}(\vee \mathcal{S})$, i.e. $K'_{\vec{x}}(\vee \mathcal{S}) \leq \vee K'_{\vec{x}}(\mathcal{S})$. Therefore $K'_{\vec{x}}(\vee \mathcal{S}) = \vee K'_{\vec{x}}(\mathcal{S})$, $K'_{\vec{x}}$ is continuous. ■

Since we have a continuous function, it is now possible to find its fixpoint. In the following three propositions we will find that and verify that $Rec(G)$ is well defined as promised.

Proposition 4.10 *If $K'_{\vec{x}}$ is the extension obtained in definition 4.7, then $K'_{\vec{x}}$ has a least fixpoint (P, P') and $(P, P') = \bigvee \{K'^m_{\vec{x}}(\{\epsilon\}, \{\epsilon\}) : m \in \mathbb{N}_0\}$.*

Proof: Directly from proposition 4.9 and from the fact that continuous functions have always fixed points. ■

Since for all $u \in P$ there exists only one $a \in \Sigma$ such that $ua \in P$, we conclude that P characterizes an ω -string ξ . Similarly, we conclude that P' characterizes an ω -string ξ' and the following result holds.

Proposition 4.11 *Let ξ, ξ' be the ω -strings characterized by the sets P, P' of proposition 4.10, $\vec{x} = (x_1, \dots, x_n) \in nZSig(\Sigma)^n$ and τ be a time scale such that*

$$\text{if } t \leq t', \text{ then } t \in \tau \text{ iff } \exists i, x_i \text{ is not constant at } t,$$

for $t' \in \mathbb{R}^+$ such that \vec{x} is not constant at t' and such that \vec{x} is constant at τ for $\tau > t'$. Then the non-Zeno signal characterized by ξ, ξ', τ is a fixpoint of $H_{\vec{x}}$.

Proof: Since ξ, ξ' are characterized by P, P' respectively and P, P' is a fixpoint of $K'_{\vec{x}}$, we have that $K_{\vec{x}}(\xi, \xi') = (\xi, \xi')$. This function $K_{\vec{x}}$ is a characterization of $H_{\vec{x}}$ as we have seen in proposition 4.3 and then, if ξ, ξ', τ characterizes z , $K_{\vec{x}}(\xi, \xi'), \tau$ characterizes $H_{\vec{x}}(z)$. But $K_{\vec{x}}(\xi, \xi') = (\xi, \xi')$, therefore $H_{\vec{x}}(z) = z$, i.e., z is a fixpoint of $H_{\vec{x}}$. ■

Proposition 4.12 *$Rec(G)$ is well defined.*

Proof: Given $\vec{x} \in nZSig(\Sigma)^n$ and if we look at the definition 4.2, we verify that the operator Rec constructs the least fix point found in proposition 4.11. Therefore it is well defined since there exists only one least fixpoint when we fix the scale τ as we did in proposition 4.11. ■

About retrospectivity and memory finiteness of $Rec(G)$ we leave the following result.

Proposition 4.13 *Let G be in the conditions of definition 4.2, then $Rec(G)$ is a finite memory retrospective operator.*

Proof: Looking at definition 4.2 we can see that G is retrospective and therefore $Rec(G)$ is obviously retrospective. If $F = Rec(G)$ and $t = 0$, then we get:

$$\begin{aligned} Res(F, \vec{x}, t)(\vec{z})(\tau) &= F(\vec{x})(t + \tau) \\ &= G(x_1, \dots, x_n, Const_a)(\tau). \end{aligned}$$

If $t > 0$, then we have:

$$\begin{aligned} &Res(F, \vec{x}, t)(\vec{z})(\tau) \\ &= F((x_1]_t^t; z_1), \dots, (x_n]_t^t; z_n))(t + \tau) \\ &= G((x_1]_t^t; z_1), \dots, (x_n]_t^t; z_n), (F((x_1]_t^t; z_1), \dots, (x_n]_t^t; z_n))]^{t'}; Const_a))(t + \tau) \\ &= G((x_1]_t^t; z_1), \dots, (x_n]_t^t; z_n), (((F\vec{x})]_t^t; Res(F, \vec{x}, t)\vec{z})]^{t'}; Const_a))(t + \tau) \\ &= G((x_1]_t^t; z_1), \dots, (x_n]_t^t; z_n), ((F\vec{x})]_t^t; (Res(F, \vec{x}, t)\vec{z})]^{t'-t}; Const_a))(t + \tau) \\ &= Res(G, (x_1, \dots, x_n, F\vec{x}), t)(z_1, \dots, z_n, (Res(F, \vec{x}, t)\vec{z})]^{t'-t}; Const_a)(\tau), \end{aligned}$$

where $t' < t + \tau$, i.e. $t' - t < \tau$.

As $Res(G, (x_1, \dots, x_n, F\vec{x}), t)$ is a finite memory retrospective operator, by proposition 4.12, the number of residuals of F and G is the same, i.e., F is a finite memory operator. ■

We may now give the complete definition of $[\mathcal{X}[\Sigma_1, \dots, \Sigma_m], OP]$.

Definition 4.14 Given $\Sigma_1, \dots, \Sigma_m$ finite non-empty sets, the function algebra $[\mathcal{X}[\Sigma_1, \dots, \Sigma_m]; OP]$ is defined as follows:

$$\begin{aligned} OP &= \{\circ, Rec\}, \\ \mathcal{X}[\Sigma_1, \dots, \Sigma_m] &= \{LLim_a : a \in \Sigma_i, i = 1, \dots, m\} \\ &\quad \cup \{LJV_{k,a}^n : a \in \Sigma_i, i = 1, \dots, m, k \leq n\} \\ &\quad \cup \{P_f : n \in \mathbb{N}, f : \Sigma_i^n \longrightarrow \Sigma'_j, i, j = 1, \dots, m\} \end{aligned}$$

In the above definition, \circ and Rec are in fact sets of operations since they may be applied to operators with different arities.

The following important result holds.

Theorem 4.15 *If F is in $[\mathcal{X}[\Sigma_1, \dots, \Sigma_m]; OP]$, then F is a finite memory retrospective operator.*

Proof: Directly from the above propositions. ■

5 Circuits

If we consider the set of circuits obtained from the function algebra defined in the previous section, we know that all representable operators are finite memory retrospective operators by theorem 4.15. Now we will prove that all of them can be represented by these circuits.

Before we note that we will use the operation Rec with \vec{x} such that each x_i may be defined over different alphabets. If we look at the above section, we easily verify that the results continue to be true, in fact they are independent of the alphabets.

Theorem 5.1 *If $G : nZSig(\Sigma)^n \longrightarrow nZSig(\Sigma)$ is a finite memory retrospective operator, then there is a circuit obtained from the function algebra $[\mathcal{X}[\Sigma_1, \dots, \Sigma_m]; OP]$ that represents G .*

Proof: Given a finite memory retrospective operator $G : nZSig(\Sigma)^n \longrightarrow nZSig(\Sigma)$, let $\{G_0, \dots, G_k\}$ with $G_0 = G$ be the set of its residuals. Let $\vec{x} \in nZSig(\Sigma)^n$ and $t, t' \in \mathbb{R}^+$ such that $0 < t' < t$ and \vec{x} is constant in $]t', t[$. If G_p is the residual of G with respect to \vec{x} and t' , then, by definition 2.4, the residual of G with respect to \vec{x} and t is given by

$$\begin{aligned} & \lambda z_1 \dots z_n \lambda t'' . G((x_1]^{t'}; z_1), \dots, (x_n]^{t'}; z_n))(t + t'') \\ &= \lambda z_1 \dots z_n \lambda t'' . G_p((Jump_{a_1 \rightarrow b_1}]^{t'}; z_1), \dots, (Jump_{a_n \rightarrow b_n}]^{t'}; z_n))(t + t'') \\ &= Res(G_p, (Jump_{a_1 \rightarrow b_1}, \dots, Jump_{a_n \rightarrow b_n}), t), \end{aligned}$$

where $\vec{a} = (a_1, \dots, a_n)$ and $\vec{b} = (b_1, \dots, b_n)$ are such that $x_i(\tau) = Jump_{a_i \rightarrow b_i}(\tau)$, for $\tau \in [t', t[$ and $0 \leq i \leq n$. So, given $\vec{x} \in nZSig(\Sigma)^n$ and $t \in \mathbb{R}^+$, we obtain all residuals of G with respect \vec{x} and t using the previous residuals. Therefore, we define the function $g : \Sigma^{2n} \times \{0, \dots, k\} \longrightarrow \{0, \dots, k\}$ as follows:

$$\begin{aligned} g(b_1, \dots, b_n, a_1, \dots, a_n, p) &= q \\ \text{iff} \\ Res(G_p, Jump_{a_1 \rightarrow b_1}, \dots, Jump_{a_n \rightarrow b_n}, t) &= G_q, \end{aligned}$$

where $t > 0$.

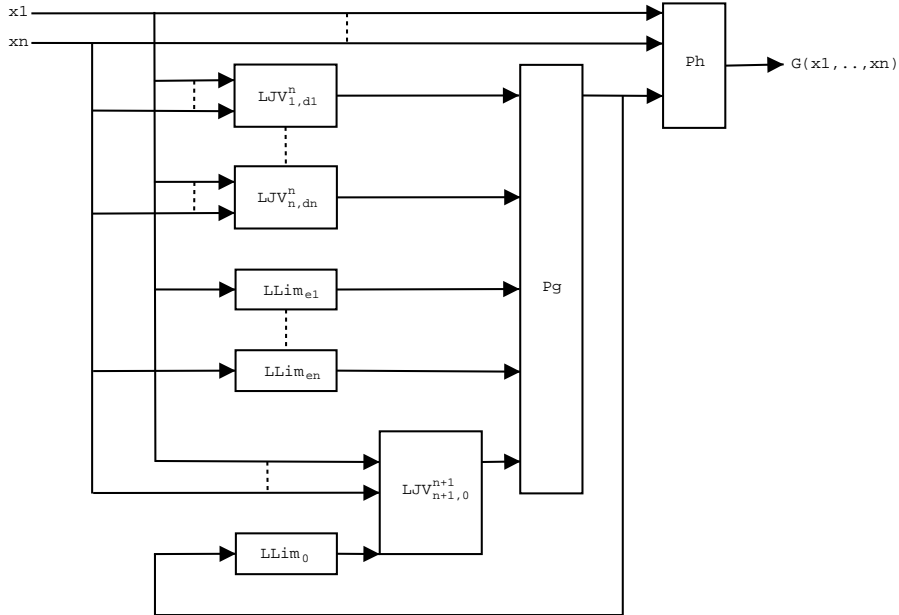


Figure 2: Construction schema for theorem 5.1. The construction reflects the primitive operators, the operator composition and the application of the *Rec* operator founded in equation 1.

Making use of the function algebra $[\mathcal{X}[\Sigma_1, \dots, \Sigma_m]; OP]$, we can define

$$\begin{aligned} J_g(\vec{x}, r) &= P_g(LLim_{e_1}(x_1), \dots, LLim_{e_n}(x_n), \\ &\quad LJV_{1,d_1}^n(\vec{x}), \dots, LJV_{n,d_n}^n(\vec{x}), \\ &\quad LJV_{n+1,0}^{n+1}(\vec{x}, LLim_0(r))) \end{aligned}$$

with \vec{e} and \vec{d} such that $g(d_1, \dots, d_n, e_1, \dots, e_n, 0) = 0$ ⁸. Looking at the definitions 3.1 and 3.2, we verify that J_g is in the conditions for the application of operator *Rec*. Given $\vec{x} \in nZSig(\Sigma)^n$, we have

$$\begin{aligned} Rec(J_g)(\vec{x})(t) &= P_g(LLim_{e_1}(x_1), \dots, LLim_{e_n}(x_n), \\ &\quad LJV_{1,d_1}^n(\vec{x}), \dots, LJV_{n,d_n}^n(\vec{x}), \\ &\quad LJV_{n+1,0}^{n+1}(\vec{x}, LLim_0(Rec(J_g)\vec{x}))(t)) \\ &= g(LLim_{e_1}(x_1)(t), \dots, LLim_{e_n}(x_n)(t), \\ &\quad LJV_{1,d_1}^n(\vec{x})(t), \dots, LJV_{n,d_n}^n(\vec{x})(t), \\ &\quad LJV_{n+1,0}^{n+1}(\vec{x}, LLim_0(Rec(J_g)(\vec{x}))(t))). \end{aligned}$$

If $t' \in [0, t[$ is such that x_i is constant in $]t', t[$, for all i , and x_j is not continuous at t' for some j , then, for each i , $x_i(t) = x_i]^{t'}$; $Jump_{LJV_{i,d_i}^n(\vec{x}) \rightarrow LLim_{e_i}(x_i)}(t)$ and $G_{LJV_{n+1,0}^{n+1}(\vec{x}, LLim_0(Rec(J_g)(\vec{x}))(t))}$ is the residual of G with respect to \vec{x} and t' . Therefore, by definition of g , $G_{Rec(J_g)(\vec{x})(t)} = Res(G, \vec{x}, t)$.

Let $0 \leq j \leq n$, G_j is a retrospective operator, then to obtain $G_j(\vec{x})(0)$ we only need to know $\vec{x}(0)$ and we get $G_i(\vec{x})(0) = G_i(Const_{a_1}, \dots, Const_{a_n})(0)$. Defining $h : \Sigma^n \times \{0, \dots, k\} \rightarrow \Sigma$ as $h(\vec{a}, j) = b$ iff $G_j(Const_{a_1}, \dots, Const_{a_n})(0) = b$, we have

$$G(\vec{x})(t) = P_h(x_1, \dots, x_n, Rec(J_g)(\vec{x}))(t). \quad (1)$$

This is the circuit of G which is exemplified in figure 2. ■

Theorems 4.15 and 5.1 taken together state that the function algebra $[\mathcal{X}[\Sigma_1, \dots, \Sigma_m]; OP]$ coincide precisely with the set of finite memory retrospective operators, i.e., $[\mathcal{X}[\Sigma_1, \dots, \Sigma_m]; OP] = \mathcal{O}_{FMR}$.

6 Examples

In this section we will construct circuits for three finite memory retrospective operators. We will use the function algebra $[\mathcal{X}[\Sigma_1, \dots, \Sigma_n], OP]$, however the general construction schema provided in theorem 5.1 will not be followed⁹.

Example 6.1 $Const_b \in nZSig(\Sigma)$

$$Const_b(t) = b$$

$Const_b$ is obviously a 0-ary finite memory retrospective operator with only one residual, $Const_b$. Therefore it is possible to represent this operator by the circuit of figure 3, i.e.,

⁸To guarantee this requisite we introduce if needed new symbols in the domain of g , that will not affect the result.

⁹The circuits obtained by the general construction schema become more complex.

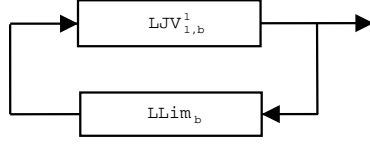


Figure 3: $Const_b$ circuit.

$$Const_b = Rec(\lambda r. LJV_{1,0}^1(LLim_0(r))).$$

Example 6.2 $Jump_{a \rightarrow b} \in nZSig(\Sigma)$

$$Jump_{a \rightarrow b}(t) = \begin{cases} a & \text{if } t = 0 \\ b & \text{if } t > 0 \end{cases}$$

In order to get a circuit for $Jump_{a \rightarrow b}$ is enough compose the circuit of figure 3 with $LLim_a$ as we did in figure 4. Therefore,

$$Jump_{a \rightarrow b} = LLim_a(Rec(\lambda r. LJV_{1,0}^1(LLim_0(r)))).$$

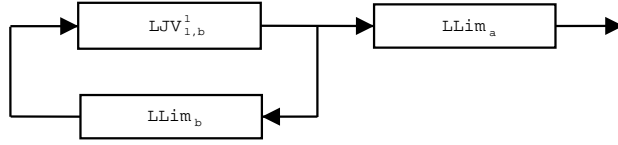


Figure 4: $Jump_{a \rightarrow b}$ circuit.

Example 6.3 $LeftCont : nZSig(\Sigma) \longrightarrow nZSig(\{True, False\})$

$$LeftCont(x)(t) = \begin{cases} True & \text{if } x \text{ is left continuous at } t \\ False & \text{otherwise.} \end{cases}$$

We assume that a signal is not continuous at 0. Clearly we can get this operator

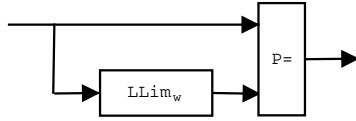


Figure 5: $LeftCont$ circuit.

as follows:

$$LeftCont(x) = P_=(x, LLim_w(x))$$

where w is a symbol such that $w \notin \Sigma$ and $P_=-$ is the pointwise extension of

$$=: (\Sigma \cup \{w\})^2 \longrightarrow \{True, False\}$$

defined as

$$=(a, b) = \begin{cases} True & \text{if } a = b \\ False & \text{otherwise.} \end{cases}$$

Therefore, in figure 5, we give a circuit to *LeftCont*.

Example 6.4 $JumpSum : nZSig(\{0, 1\}) \longrightarrow nZSig(\{0, 1\})$

$$JumpSum(x)(t) = \sum_{i=1}^n x(t_i)$$

where the t_i 's are the jumps before t and the sum is in \mathbb{Z}_2 . We can get this

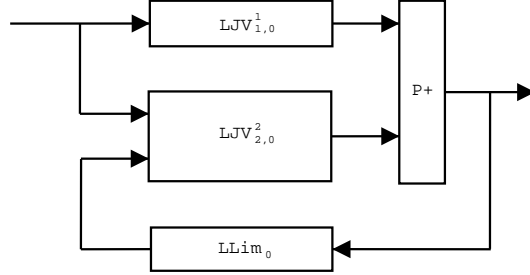


Figure 6: *JumpSum* circuit.

operator as follows:

$$JumpSum(x) = Rec(\lambda xr. P_+(LJV_{1,0}^1(x), LJV_{2,0}^2(x, LLim_0(r))))(x)$$

where P_+ is the pointwise extension of

$$+ : \{0, 1\} \longrightarrow \{0, 1\}$$

defined as expected. In figure 6 we give a circuit to *JumpSum*.

7 Automata

In this section we state the relation between the circuits introduced in previous sections and the finite state transducers introduced in [Rab97].

As we have done with respect to circuits, we will consider transducers with multiple input channels. We give now the definition of finite state transducer.

Definition 7.1 (Finite State Transducer [Rab97]) A finite state transducer \mathcal{A} over non-Zeno signals is a tuple $\langle Q, q_0, \Sigma_{in}, \Sigma_{out}, n, out, \delta \rangle$ such that

- Q is a finite non empty set of states,
- $q_0 \in Q$ is the initial state,
- Σ_{in} ¹⁰ is the input alphabet,
- Σ_{out} is the output alphabet,
- n is the arity of \mathcal{A} ,
- $out : Q \times \Sigma_{in}^n \longrightarrow \Sigma_{out}$ is the output function and

¹⁰We assume that all input signals are defined over the same alphabet, however multiple input alphabets may be considered without lost of generality.

- $\delta : \Sigma_{in}^{2n} \longrightarrow (Q \rightarrow Q)$ is the transition function which verifies

$$\delta(b_1, \dots, b_n, b_1, \dots, b_n) \circ \delta(a_1, \dots, a_n, b_1, \dots, b_n) = \delta(a_1, \dots, a_n, b_1, \dots, b_n).$$

It is important to note that each one of the states corresponds to a residual and each transition corresponds to a jump from a_1, \dots, a_n to b_1, \dots, b_n .

We will use a graphical representation for transducers. As we may see in figure 7, the states will be represented by nodes and the transitions will be represented by labeled arcs. The output function can be represented by labeling the nodes as follows:

$$\begin{aligned} q \text{ has label } &< (c_{11}, \dots, c_{n1})/d_1, \dots, (c_{1k}, \dots, c_{nk})/d_k > \\ &\text{if and only if} \\ &out(q, c_{1i}, \dots, c_{ni}) = d_i, \text{ for every } i. \end{aligned}$$

The initial state will be indicated with an arrow as is done in figure 7.

A transducer receives a signal $\vec{x} \in nZSig(\Sigma_{in})^n$ as input and produces as output a signal $y \in nZSig(\Sigma_{out})$. If $\vec{x}(t) = (a_1, \dots, a_n)$, for $t \in \mathbb{R}^+$, then $y(t) = b$ where b is given accordingly with the label of the current node, i.e., accordingly with the *out* function in each state. A transition between states occurs accordingly with the jump of \vec{x} at t , if \vec{x} produces a jump from (a_1, \dots, a_n) to (b_1, \dots, b_n) at t , then occurs a transition from the current state to other one, where the transition arc is labeled by $< (a_1, \dots, a_n), (b_1, \dots, b_n) >$.

Definition 7.2 (Operator Computable by a Transducer)

Let $\mathcal{A} = \langle Q, q_0, \Sigma_{in}, \Sigma_{out}, n, out, \delta \rangle$ be a transducer. The operator $F_{\mathcal{A}}$ computable by \mathcal{A} is defined as $out(F^\delta \vec{x} t q_0, \vec{x} t)$, where F^δ is the unique operator definable by δ [Rab97].

In [Rab97] has been stated that finite state transducers compute finite memory retrospective operators over non-Zeno signals and that the inverse also holds. Therefore the class of operators computed by finite state transducers is exactly the same class of operators characterized by the circuits introduced before.

We provide now an example.

Example 7.3 The unary operator

$$LLim_0 : nZSig(\{0, 1\}) \longrightarrow nZSig(\{0, 1\})$$

is defined as follows:

$$LLim_0(x)(t) = \begin{cases} 0 & \text{if } t = 0 \\ b & \text{if } t > 0 \text{ and } b \in \{0, 1\} \text{ is the left limit of } x \text{ at } t. \end{cases}$$

This operator have two residuals:

$$\begin{aligned} LLim_0(z)(\tau) &= \begin{cases} 0 & \text{if } \tau = 0 \\ b & \text{if } \tau > 0 \text{ and } b \in \{0, 1\} \text{ is the left limit of } z \text{ at } \tau, \end{cases} \\ LLim_1(z)(\tau) &= \begin{cases} 1 & \text{if } \tau = 0 \\ b & \text{if } \tau > 0 \text{ and } b \in \{0, 1\} \text{ is the left limit of } z \text{ at } \tau. \end{cases} \end{aligned}$$

Thus, we construct the transducer $\mathcal{A}_{LLim_0} = \langle Q, q_0, \Sigma_{in}, \Sigma_{out}, n, out, \delta \rangle$.

- Since there are two residuals, we need two states, i.e, $Q = \{q_0, q_1\}$. Each state is associated to a residual as follows:

$$q_0 \dashrightarrow LLim_0 \text{ and } q_1 \dashrightarrow LLim_1.$$

- The initial state is the state associated to the operator $LLim_0$, i.e., q_0 .
- $\Sigma_{in} = \{0, 1\}$.
- $\Sigma_{out} = \{0, 1\}$.
- Since $LLim_0$ is an unary operator, we have $n = 1$.
- We obtain the function *out* as follows:

$$\begin{aligned} out(q_0, 0) &= LLim_0(Const_0)(0) = 0 \\ out(q_0, 1) &= LLim_0(Const_1)(0) = 0 \\ out(q_1, 0) &= LLim_1(Const_0)(0) = 1 \\ out(q_1, 1) &= LLim_1(Const_1)(0) = 1 \end{aligned}$$

State	Transition	δ	<i>out</i>
q_0	0, 0	q_0	$0 \mapsto 0$
q_0	0, 1	q_1	$0 \mapsto 0$
q_0	1, 0	q_0	$1 \mapsto 0$
q_0	1, 1	q_1	$1 \mapsto 0$
q_1	0, 0	q_0	$0 \mapsto 1$
q_1	0, 1	q_1	$0 \mapsto 1$
q_1	1, 0	q_0	$1 \mapsto 1$
q_1	1, 1	q_1	$1 \mapsto 1$

Table 1: δ and *out* functions in \mathcal{A}_{LLim_0} .

- Since Σ_{in} has two symbols, we know that there exist four different jumps:

$$Jump_{0 \rightarrow 0}, Jump_{0 \rightarrow 1}, Jump_{1 \rightarrow 0} \text{ and } Jump_{1 \rightarrow 1},$$

i.e., there exist four possible transitions at each state. Therefore, the function δ is defined as follows:

$$\begin{aligned} \delta(0, 0)q_0 &= q_0 \text{ because } Res(LLim_0, Jump_{0 \rightarrow 0}, t) = LLim_0, \\ \delta(0, 0)q_1 &= q_0 \text{ because } Res(LLim_1, Jump_{0 \rightarrow 0}, t) = LLim_0, \\ \delta(0, 1)q_0 &= q_1 \text{ because } Res(LLim_0, Jump_{0 \rightarrow 1}, t) = LLim_1, \\ \delta(0, 1)q_1 &= q_1 \text{ because } Res(LLim_1, Jump_{0 \rightarrow 1}, t) = LLim_1, \\ \delta(1, 0)q_0 &= q_0 \text{ because } Res(LLim_0, Jump_{1 \rightarrow 0}, t) = LLim_0, \\ \delta(1, 0)q_1 &= q_0 \text{ because } Res(LLim_1, Jump_{1 \rightarrow 0}, t) = LLim_0, \\ \delta(1, 1)q_0 &= q_1 \text{ because } Res(LLim_0, Jump_{1 \rightarrow 1}, t) = LLim_1, \\ \delta(1, 1)q_1 &= q_1 \text{ because } Res(LLim_1, Jump_{1 \rightarrow 1}, t) = LLim_1, \end{aligned}$$

for $t > 0$.

In figure 7 we give the transducer \mathcal{A}_{LLim_0} .

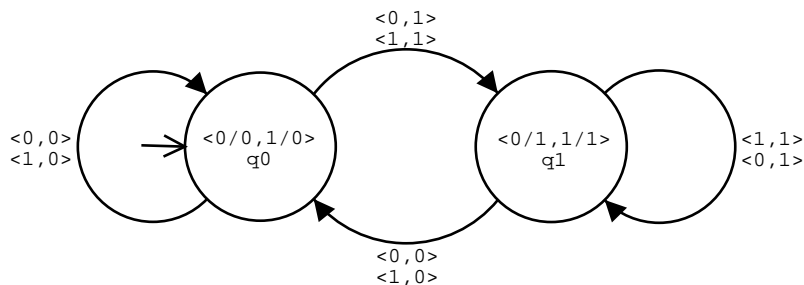


Figure 7: Transducer \mathcal{A}_{LLim_0} .

8 Final Remarks

Circuits of operators were introduced. The concept of function algebra in [Clo99] was used in order to obtain an algebra of finite memory retrospective operators. Our main contribution is the proof of the equivalence between this algebra of operators and the set of finite memory retrospective operators.

It is important to note that we did not use time delay operators, which are commonly used in the classical theory of circuits. In order to know the values of signals at previous instants we used the $LLim$ operator and the LJV operator and with these operators we have proved that it is possible construct circuits for any finite memory retrospective operator.

Examples of circuits for some operators were also included. However not used for these examples, the general construction schema provided in the proof of theorem 5.1 permits the construction of circuits for any finite memory retrospective operator using the function algebra $[\mathcal{X}[\Sigma_1, \dots, \Sigma_n], OP]$.

The research around automata over continuous time, started in [Tra98] and [Rab97], was expanded in this article, where we succeed to develop a theory of circuits of retrospective operators to fully characterize the class of finite memory retrospective operators. However many other open problems are to be solved and the field to be enlarged towards a theory on foundations of hybrid systems. Boaz Trakhtenbrot gave us a copy of his personal notes, where many interesting problems were designed, e.g., the problem of oracles (the relativization problem), a theory of reducibilities between operators has been considered and sketched, a theory of reliable feedback has been developed in some depth, etc. We thus hope that such a foundational seminal work prosper in the near future.

9 Acknowledgments

This paper is a further development of [Rab97], encouraged by Alexander Rabinovich when he visited the LabMAC (*Laboratório de Modelos e Arquiteturas Computacionais, FCUL*) and the CLC (Center for Logic and Computation, IST) at Lisbon. The author is very grateful to him and also to the father of the theory of automata over continuous time – B. Trakhtenbrot. The author would also like to thank his advisor J. Félix Costa for his guidance and critical analysis.

This work was the main reason for the integration of the author in the project ConTComp, an initiative of CLC partially supported by FCT (*Fundação para*

a Ciência e a Tecnologia) and EU FEDER. The visit of Alexander Rabinovich was supported by LabMAC.

References

- [AD94] R. Alur and D. L. Dill. A Theory of Timed Automata. *Theoretical Computer Science*, 126:183–235, 1994.
- [Clo99] P. Clote. Computational models and function algebras. In E. R. Griffor, editor, *Handbook of Computability Theory*, pages 589–681. Elsevier, 1999.
- [KT65] N. E. Kobrinskii and B. A. Trakhtenbrot. *Introduction to the Theory of Finite Automata*. North-Holland Publishing Company Amsterdam, 1965.
- [PRT01] D. Pardo, A. Rabinovich, and B. A. Trakhtenbrot. Synchronous Circuits over Continuous Time: Feedback Reliability and Completeness. Technical report, School of Computer Science, Tel Aviv University, Ramat Aviv, Tel Aviv 69978, Israel, 2001.
- [Rab97] A. Rabinovich. Automata over Continuous Time. To appear in *Theoretical Computer Science* in print, 1997.
- [RT98] A. Rabinovich and B. A. Trakhtenbrot. From Finite Automata toward Hybrid Systems (Extended Abstract). In *FCT: Fundamentals (or Foundations) of Computation Theory*, 1998.
- [Son90] E. D. Sontag. *Mathematical Control Theory — Deterministic Finite Dimensional Systems*, volume 6 of *Texts in Applied Mathematics*. Springer-Verlag, June 1990.
- [Tra98] B. Trakhtenbrot. Automata and Hybrid Systems. Technical Report 153, Uppsala University, February 1998.
- [Tra99] B. A. Trakhtenbrot. Automata and Their Interaction: Definitional Suggestions. In *Fundamentals of Computation Theory*, pages 54–89, 1999.