(Co)Algebraic Characterizations of Signal Flow Graphs

Henning Basold[1,3], Marcello Bonsangue[2,3], Helle H. Hansen[1,3], Jan Rutten[3,1]

[1] Radboud University Nijmegen
[2] Leiden University
[3] CWI Amsterdam

Abstract One of the first publications of Prakash Panangaden is about compositional semantics of digital networks, back in 1984. Digital networks transform streams of input signals to streams of output signals. If the output streams of the components of the network are functions of their input streams, then the behavior of the entire network can be nicely characterized by a recursive stream function. In this paper we consider signal flow graphs, i.e., open synchronous digital networks with feedbacks, obtained by composing amplifiers, mergers, copiers, and delayers. We give two characterizations of the recursive stream functions computed by signal flow graphs: one algebraic in terms of localization of modules of polynomials, and another coalgebraic in terms of Mealy machines. Our main result is that the two characterizations coincide.

"Tell all the truth but tell it slant - success in circuit lies."
— Emily Dickinson

1 Introduction

Signal flow graphs are a graphical representation for the analysis, modeling and evaluation of linear systems as studied, for example, in signal processing algorithms and systems theory [6,20]. They also occur in the literature as linear flow graphs [16], linear finite state machines [17] or linear (sequential) circuits [25,18]. Here we will also refer to them simply as *circuits* since they are a special case of digital networks.

Signal flow graphs are directed graphs in which the nodes transform input signals into output signals (by amplifying, copying and merging), the arcs communicate signals without delay [12,13] (unlike in data flow graphs [11,7]), and signal delay is implemented by registers. An arc which connects only a node at its target is called an input end, similarly, an output end connects only a node at its source. We classify signal flow graphs, along two parameters: being open/closed and feedforward/feedback, where a signal flow graph is *open* if it has an input end, otherwise it is *closed*. A signal flow graph is *feedforward* if it has no cycles; otherwise it is a *feedback* circuit. Our main object of study are open, feedback circuits. All other cases are viewed as instances of them.

The behavior of a signal flow graph can be nicely characterized by a recursive stream function. This works as follows. The internal state of the circuit is its register contents. The internal state in the next step can be calculated as a linear combination of the input and the registers. That is, the dynamics of a circuit can be expressed as a system of linear of equations, one for each register. Since we consider open circuits, the corresponding linear systems may have more variables than equations where these extra (read-only) variables correspond to the input arcs of the circuit.

One way of associating a stream transformation to an open linear system is to solve it *algebraically.* Assuming that signals are elements of a unital, commutative ring R, we present a generalization of the method given in [23] for solving *closed* linear systems. The solution of a closed linear system is a fraction of polynomials (i.e., a rational stream [24]). The solution of an open linear system is also a fraction of polynomials, but its numerator consists of two or more polynomials: one represents the initial register contents, and the others represent the dependency on the input arcs. More precisely, we show that the solutions of open linear systems are characterized by the localization of free modules of polynomials over R.

Our second observation is that open linear systems (over a unital commutative ring R) can be viewed as (generally infinite) Mealy machines with input and output in R, and a free R-module as state space. Since Mealy machines are coalgebras [3,9], they come equipped with a unique behavior map associating each state of a Mealy Machine to a causal function from (input) streams to (output) streams. In this way we obtain a *coalgebraic* characterization of signal flow graphs.

Our main result is to show that the algebraic and the coalgebraic characterizations of linear systems coincide. As a consequence we obtain a novel sound and complete axiomatization of signal flow graphs, as well as an effective procedure for deciding circuit equivalence.

Related work. A strictly smaller class of signal flow graphs and their behaviors has already been studied coalgebraically in [23,24], where the behaviors of closed feedback circuits with signals from a field are characterized as rational streams. In fact, our method for computing stream transformations for open feedback circuits is a generalization of the one in [23,24] for computing streams. Also in [23], the behaviors of open feedback circuits in which all registers are initially 0 were characterized as stream functions that multiply the input stream by a fixed rational stream, but no algebraic characterization was provided. An alternative algebraic calculus (without polynomials but using fixed points) for closed feedback circuits with signals over a field is given in [19], which yields also a sound and complete axiomatization of rational streams. An extension of the latter calculus (again without polynomials but using fixed points) to weighted automata over alphabets of arbitrary size and weights in a semiring is presented in [4]. Our method to represent stream transformations by fractions of polynomials over two or more generators is inspired by the work in [10].

Finally, we mention the classical methods of finding closed forms for linear recurrences [28], which correspond to closed systems. A well-known example is the formula for computing the Fibonacci numbers, involving the golden ratio. There is also work (e.g., in [25]) on finding closed forms for what we call an open system here, i.e., linear recurrences with a parameter. Such closed forms allow efficient computation of single values, but usually it is difficult to check for equivalence of closed forms. This differs from our approach which yields methods for effective comparison, but not so much efficient computation of single values.

Overview. In Section 2 we recall basic facts from ring theory and universal coalgebra. Signal flow graphs and their relationship with open linear systems are briefly discussed in Section 3. In Section 4 we present the relevant algebraic structures needed to solve open linear systems, and show their correspondence with subsets of causal functions. Open linear system are solved algebraically in Section 5, and coalgebraically in Section 6. The two solutions are shown to coincide in Section 7. We summarize our results, and discuss future directions in Section 8.

2 A bit of algebra and coalgebra

In this section we recall the basic material from ring theory and from the coalgebraic stream calculus needed throughout the paper. A more extensive introduction to commutative ring theory can be found in [1,8]. For the stream calculus and universal coalgebra we refer to [21,23].

2.1 Rings, modules and algebras

Throughout this paper we let R denote a *unital commutative ring*, that is, a ring $(R, +, \cdot, 0, 1)$ in which the multiplication is commutative with neutral element 1. Furthermore, we assume R is *non-trivial*, i.e., $0 \neq 1$ in R. We call an element $a \in R$ *invertible* if it has a multiplicative inverse $a^{-1} \in R$, i.e., $a \cdot a^{-1} = 1 = a^{-1} \cdot a$. We denote by $R^{\times} \subseteq R$ the set of invertible elements of R. If for $a, b \in R$ the inverses a^{-1}, b^{-1} exist, then $(ab)^{-1} = b^{-1}a^{-1}$.

A *(unital) R-module* is an abelian group $(M, +, 0)$ together with a scalar multiplication, written am for $a \in R$ and $m \in M$, such that for every $a, b \in R$ and every $m, n \in M$ the following identities hold:

$$\begin{aligned} (a+b)m &= am + bm \qquad & a(m+n) &= am + an \\ 1m &= m & a(bm) &= (ab)m\,. \end{aligned}$$

Both rings and modules come with the usual notion of homomorphism. Module homomorphisms will also be called *linear maps*.

A map $f\colon S \to R$ is said to have *finite support* if $f(x) \neq 0$ only for finitely many elements of $x \in S$. For every set S, the *free R-module* over S exists and can be constructed as the set R^S of all maps in $S \to R$ with finite support. Addition and scalar multiplication are defined point-wise. Often we write an element $m \in$

R^S as a linear combination $m = a_1x_1 + \cdots + a_nx_n$, where $x_1, \ldots, x_n$ are the support of m. By universality of free constructions, every function $f\colon S \to M$, where M is an R-module, can be uniquely extended to a linear map $\overline{f}\colon R^S \to M$ such that $f = \overline{f} \circ i$, where $i(x) = 1x$ is the inclusion of S into R^S. The extension of f is given by $\overline{f}(a_1x_1 + \cdots + a_nx_n) = a_1f(x_1) + \cdots + a_nf(x_n)$.

A subset V of a module M is *linearly independent* if whenever $\sum_{i=1}^{k} a_iv_i = 0$ for some $a_i \in R$ and $v_i \in V$, then we have $a_i = 0$ for all $1 \leq i \leq k$. If $f\colon S \to M$ is an injective map into an R-module M such that $f(S)$ is a linearly independent subset of M, then $\overline{f}\colon R^S \to M$ is injective.

If R is commutative and non-trivial, then finitely generated free modules behave like vector spaces over a field. In fact, every generator set of R^S has the same cardinality as S. In this case the ring R is said to have an *invariant basis number* and S is called a *basis* for the free module. Having such a basis, linear maps between free finitely generated R-modules can be seen as matrices.

An R-*algebra* is an R-module that is also a commutative ring having a multiplication that is bilinear. Equivalently, an R-algebra is a pair (A, ψ) such that A is a ring and $\psi\colon R \to A$ is a ring homomorphism.

For example, every ring R is trivially an R-algebra, and hence an R-module. A prototypical example of an R-algebra is the ring of polynomials $R[X]$ in a single variable X (with the inclusion $a \mapsto aX^0$).

2.2 Coalgebras

Given a functor $F\colon \mathbf{Set} \to \mathbf{Set}$ on the category of sets and functions, an F-*coalgebra* consists of a set S together with a *structure map* $c\colon S \to FS$. An F-coalgebra *homomorphism* $f\colon (S_1, c) \to (S_2, d)$ is a map $f\colon S_1 \to S_2$ such that $d \circ f = F(f) \circ c$. The F-coalgebras together with their homomorphisms form a category denoted by $\mathrm{Coalg}(F)$. A *subcoalgebra* of an F-coalgebra (S_1, c_1) is an F-coalgebra (S_2, c_2) if the inclusion map $S_2 \hookrightarrow S_1$ is a homomorphism.

An F-coalgebra (Z, ζ) is said to be *final* if for any F-coalgebra (S, c) there exists a unique homomorphism $\widetilde{c}\colon (S, c) \to (Z, \zeta)$. The carrier Z can be thought of as the set of all *observable behaviors* of F-coalgebras, and the unique homomorphism $\widetilde{c}\colon (S, c) \to (Z, \zeta)$ is therefore also called the *behavior map*. A final F-coalgebra, if it exists, is unique up to isomorphism. The structure map ζ of a final coalgebra is necessarily an isomorphism [15].

An F-*bisimulation* between two F-coalgebras (S_1, c_1) and (S_2, c_2) is a relation $B \subseteq S_1 \times S_2$ that can be equipped with an F-coalgebra structure b such that both projections $\pi_1\colon B \to S_1$ and $\pi_2\colon B \to S_2$ are F-coalgebra homomorphisms. Two elements $s_1 \in S_1$ and $s_2 \in S_2$ are *bisimilar* if there exists an F-bisimulation B containing the pair (s_1, s_2). We will use bisimulations as a tool for proving that two states have the same observable behavior.

Proposition 1. *Let (S_1, c_1) and (S_2, c_2) be two F-coalgebras with $s_1 \in S_1$ and $s_2 \in S_2$, and assume that a final F-coalgebra exists. If s_1 and s_2 are bisimilar, then $\widetilde{c_1}(s_1) = \widetilde{c_2}(s_2)$, i.e., they have the same behavior.*

The converse of the above proposition holds under the assumption that the functor F preserves weak pullbacks [21].

2.3 Elements of stream calculus

For a ring R, coalgebras for the functor $\mathcal{S} = R \times (-)$ are called *stream automata*. The final $\mathcal{S}$-coalgebra is given by the set of all *streams* $\sigma \in R^\omega = \mathbb{N} \to R$ together with the structure map ξ defined by $\xi(\sigma) = (\sigma(0), \sigma')$, where $\sigma(0)$ is the *initial value* of σ is σ' its *derivative* [22]), i.e., for all $n \in \mathbb{N}$, $\sigma'(n) = \sigma(n + 1)$. The inverse of ξ is given by the *cons* map defined by $(r : \sigma)(0) = r$, $(r : \sigma)(n{+}1) = \sigma(n)$ for all $n \in \mathbb{N}$.

We define operations on R^ω by means of *behavioral differential equations* [22], i.e., by specifying their initial value and derivative. The following operations become relevant for the algebraic characterizations of circuit behaviors in Section 5.

Initial value	Derivative	Name
$[r](0) = r$	$[r]' = [0]$	constant
$X(0) = 0$	$X' = [1]$	shift
$(\sigma + \tau)(0) = \sigma(0) + \tau(0)$	$(\sigma + \tau)' = \sigma' + \tau'$	sum
$(\sigma \times \tau)(0) = \sigma(0) \cdot \tau(0)$	$(\sigma \times \tau)' = \sigma' \times \tau + [\sigma(0)] \times \tau'$	convolution product
$\sigma^{-1}(0) = \sigma(0)^{-1}$	$(\sigma^{-1})' = [-\sigma(0)^{-1}] \times \sigma' \times \sigma^{-1}$	convolution inverse

In the first column, $r \in R$ and the operations $+$, $\cdot$ and $(-)^{-1}$ on the right-hand side of the equations are operations on R. We note that the inverse is only defined on streams σ for which $\sigma(0) \in R^\times$. Note that $(\sigma \times X) = (0, \sigma_0, \sigma_1, \dots)$ for all $\sigma \in R^\omega$. We will use the so-called *fundamental theorem of stream calculus* [22].

Proposition 2. *For any $\sigma \in R^\omega$ we have: $\sigma = [\sigma(0)] + (\sigma' \times X)$.*

Proof. We note that $(r : \tau) = (r, \tau_0, \tau_1, \dots) = [r] + (0, \tau_0, \tau_1, \dots) = [r] + (\tau \times X)$ for any $\tau \in R^\omega$, and thus we obtain the desired result: $\sigma = (\sigma_0 : \sigma') = [\sigma_0] + (\sigma' \times X)$.

Streams over R form a unital, commutative ring $(R^\omega, +, \times, [0], [1])$, and an R-algebra via the embedding $a \mapsto [a]$ of R into R^ω. In other words R^ω is an R-module with the scalar multiplication $a\sigma = [a] \times \sigma$. We denote by $[R^\omega, R^\omega]$ the set of all *stream transformations*, i.e., all functions from R^ω to R^ω. It forms a ring under point-wise sum and (convolution) product, as well as an R^ω-algebra via the embedding of R^ω as the subring of constant maps. A stream transformation $f \colon R^\omega \to R^\omega$ is said to be *causal* whenever the n-th output of f depends only on the elements up to n of its input. More precisely, f is causal if for all $\sigma, \tau \in R^\omega$, if $\sigma(k) = \tau(k)$ for all $k \leq n$ then $f(\sigma)(n) = f(\tau)(n)$.

Causal stream transformations play a key role in this paper. For example, all constant maps as well as the identity are causal. We let $C(R^\omega) \subseteq [R^\omega, R^\omega]$ denote the subset of causal transformations on R^ω. Since causal functions are closed under point-wise sum and point-wise convolution product, the set $C(R^\omega)$ inherits the ring structure as well as the R^ω-algebra structure from $[R^\omega, R^\omega]$.

3 Signal flow graphs and linear systems

3.1 Signal flow graphs

A *signal flow graph* [20,23] is a directed graph in which arcs connect up-to two nodes. Nodes, also called gates, perform operations on signals from incoming arcs and output the result on outgoing arcs. Signals are assumed to be elements of a unital, commutative ring R. The *amplifier gate* ($\xrightarrow{}\overset{a}{\circ}\xrightarrow{}$) performs the scalar multiplication of the incoming signal with $a \in R$. The *adder gate* ($\oplus$) outputs the sum of its inputs. A *copier* ($\bullet$) simply copies its input to two (or more) output ends. Finally, a *register* ($\rightarrow\boxed{a}\rightarrow$) is a one-element buffer which outputs its content and stores the incoming signal. The initial content of the register is thereby a. Arcs with no connecting gates at their source are *input ends*, whereas those with no connecting gates at their target are *output ends*. For clarity, input ends are marked with an input stream ι. We will also refer to signal flow graphs simply as *circuits*. For technical simplicity we will consider only circuits with at most one input ι end and one output end.

A circuit with no input end is *closed*, otherwise it is *open*. A circuit is called *feedforward* if it contains no cycles; otherwise it is a *feedback* circuit. In order for feedback circuits to have a well-defined behavior, all cycles are required to pass through at least one register. Intuitively, the reason is that otherwise we would end up with equations which may not have unique solutions. The condition will be used in the construction in Section 3.2.

By feeding signals to the input end of a circuit we observe signals on its output end. Since there is no limit on the number of signals a circuit can react to, the *behavior* of a circuit is given by a function transforming input streams to output streams. Closed circuits do not need any input, and their behavior is given by constant stream functions, or, equivalently, by streams.

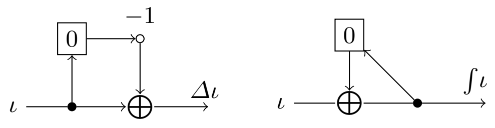


Figure 1. A differentiator and an integrator signal flow graph

For example, the leftmost circuit in Fig. 1, implements a *discrete differential* $\Delta\iota$ where, for all $n \in \mathbb{N}$,

$$(\Delta\iota)(0) = \iota(0) \quad \text{and} \quad (\Delta\iota)(n+1) = \iota(n+1) - \iota(n)\,.$$

It consists of a copier, a register with initial value 0, a multiplication by -1 and an adder. The rightmost circuit implements the *discrete integral* $\int\iota$ defined, for all $n \in \mathbb{N}$, by

$$\left(\int\iota\right)(0) = \iota(0) \quad \text{and} \quad \left(\int\iota\right)(n+1) = \iota(n+1) + \left(\int\iota\right)(n)\,.$$

Note that the discrete differential circuit is feedforward, whereas the integration circuit is a feedback circuit.

By composing the two circuits (i.e., by linking the output arc of one circuit with the input arc of the other), we obtain a new circuit that implements $\int(\Delta\iota)$. One can show by fairly straightforward induction that $\int(\Delta\iota)(n) = \iota(n)$ for all $n \in \mathbb{N}$. However, for more involved circuits, this may not always be easy. One of the applications of our results is an algebraic calculus for proving circuit equivalence (as opposed to point-wise, inductive reasoning).

3.2 Linear systems

Signal flow graphs are graphical representations of linear systems, i.e., systems of linear equations [5]. An *(n-dimensional open) linear system* $L = (V, M, O, I)$ is defined as follows. The set $V = \{v_1, \ldots, v_n\}$ is a set of n variables denoting the registers, and in addition, we have a variable v_{in} denoting the input signal. Together they form the set $\overline{V} = \{v_{\text{in}}\} + V$. We use the free module R^V to model the assignment of register contents: every element $\sum_{i=1}^n s_i v_i \in R^V$ (written as $(s_1, \ldots, s_n)$) denotes the assignment of s_i to the register v_i. Analogously, the elements of $R^{\overline{V}}$ combine the input value for v_{in} and assignments to registers. Next, M and O are linear maps $M\colon R^{\overline{V}} \to R^V$ and $O\colon R^{\overline{V}} \to R$ that describe the circuit wiring through which new values are fed to the registers and to the output end. Since M is a linear map between free modules it can be represented as an $n \times (n+1)$-matrix over R with entries $m_{i,j}$ coming from: $M(v_{\text{in}}) = \sum_{i=1}^n m_{i,0} v_i$ and $M(v_j) = \sum_{i=1}^n m_{i,j} v_i$, for $1 \le j \le n$. Similarly, O is a $1 \times (n+1)$-matrix with entries o_i given by $o_0 = O(v_{\text{in}})$, and $o_i = O(v_i)$, for $1 \le i \le n$. Together, M and O describe a system of $n+1$ equations in the variables $v_{\text{in}}, v_1, \ldots, v_n$. Finally, $I = (r_1, \ldots, r_n) \in R^V$ is the vector of initial register contents.

One should think of the register contents over time as streams. If we denote the current state by $(s_1, \ldots, s_n)$ (now viewed as a tuple of streams), the input stream by ι, the next state of the system by $(s'_1, \ldots, s'_n)$, and the output stream by o, then they satisfy the following system of stream differential equations:

$$\begin{array}{ll} \underbrace{\begin{array}{c} s_1(0) = r_1 \\ \vdots \\ s_n(0) = r_n \end{array}}_{I} & \left.\begin{array}{l} s'_1 = m_{1,0}\iota + m_{1,1}s_1 + \cdots + m_{1,n}s_n \\ \vdots \\ s'_n = m_{n,0}\iota + m_{n,1}s_1 + \cdots + m_{n,n}s_n \end{array}\right\} M \\ & \quad o \;=\; o_0\iota + \; o_1 s_1 + \cdots + \; o_n s_n \;\; \}\, O \end{array}$$

By adapting the constructions given in [23] and [19], for every signal flow graph C with one input end, one output end, finitely many gates and n registers, we define its associated n-dimensional open linear system

$$\mathcal{L}(C) = (V, M, O, I) \tag{1}$$

as follows. We set $V = \{v_1, \ldots, v_n\}$, and take $I = (r_1, \ldots, r_n) \in R^V$ to be the vector of the initial content of the registers of C. To define M, we build for each register v_i a linear combination over $v_{\text{in}}, v_1, \dots, v_n$ by traversing the graph in reverse direction starting from the input arc of v_i and ending at a register or an input end. These reverse paths form a syntax tree, say T_i, with the root being the source of the arc entering the register v_i and leaves among $v_{\text{in}}, v_1, \dots, v_n$. The tree branches at adder-gates and passes through a number of scalar multiplier gates. If a branch ends in register v_j the tree has a leaf v_j, similarly for paths ending in the input end ι, the tree has a leaf v_{in}. Since we assume all cycles pass through a register, this tree is finite. Each tree T_i gives rise to a linear combination $m_{i,0}v_{\text{in}} + m_{i,1}v_1 + \cdots + m_{i,n}v_n \in R^{\overline{V}}$ by evaluating T_i top-down in $R^{\overline{V}}$, and we define the i-th row of M (seen as a matrix) to be $M_i = (m_{i,0}, m_{i,2}, \dots, m_{i,n})$. For the output of the circuit we get again a tree, which yields a linear combination $O \in R^{\overline{V}}$ in the same way.

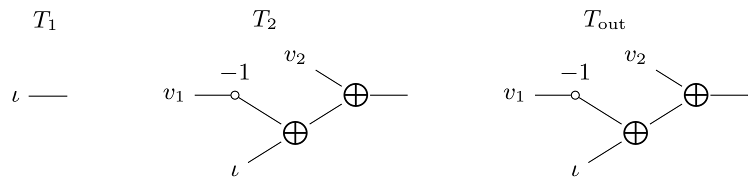


Figure 2. Trees for flow in the composition of ciruits in Fig. 1

Example 3. Let v_1, v_2 denote the two registers from left to right in the composition of the circuits given in Fig. 1 that computes $\int \Delta\iota$. The constructed trees for the registers and the output are shown in Fig. 2. These trees result in linear combinations $\iota + 0v_1 + 0v_2$, $\iota - v_1 + v_2$ and again $\iota - v_1 + v_2$, hence the linear system given by

$$V = \{v_1, v_2\} \quad I = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad M = \begin{pmatrix} 1 & 0 & 0 \\ 1 & -1 & 1 \end{pmatrix} \quad O = \begin{pmatrix} 1 & -1 & 1 \end{pmatrix}.$$

Conversely, we can construct from every linear system L a circuit, such that its associated linear system is L.

Proposition 4. *For all open linear systems L, there is a linear circuit $\mathcal{C}(L)$ such that $\mathcal{L}(\mathcal{C}(L)) = L$. In other words, transforming $\mathcal{C}(L)$ back into a linear system, see* (1)*, yields the original system L again.*

Proof. Let $L = (V, M, O, I)$ be a linear system. Figure 3 sketches the linear circuit $\mathcal{C}(L)$. In order to keep the picture simple, we use arrows pointing to and originating from v_i instead of drawing the full graph. By applying the definition, we can check that $\mathcal{L}(\mathcal{C}(L)) = L$. $\square$

The shape of the matrix M of a linear system associated to a signal flow graph gives us a precise characterization of closed and feedforward circuits.

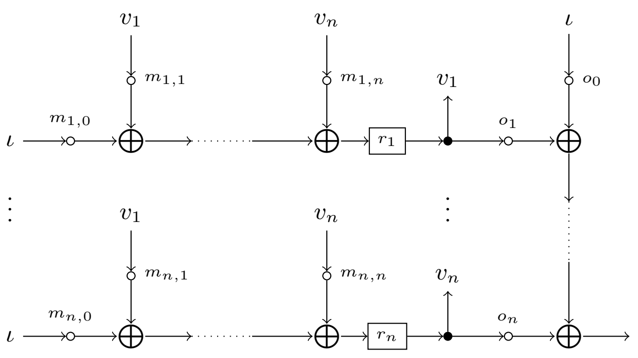


Figure 3. Signal flow graph constructed from an open linear system

Lemma 5. *Let C be a signal flow graph, and $\mathcal{L}(C) = (V, M, O, I)$ its associated linear system.*

1. *If C is closed, then the first column of M and of O contains only 0's.*
2. *If C is feedforward, then M is of the form*

$$M = \begin{pmatrix} m_{1,0} & 0 & 0 & \cdots & 0 & 0 \\ m_{2,0} & m_{2,1} & 0 & \cdots & 0 & 0 \\ & & & \vdots & & \\ m_{n,0} & m_{n,1} & m_{n,2} & \cdots & m_{n,n-1} & 0 \end{pmatrix}$$

The second part follows from the fact that, in a feedforward circuit, the register variables can be ordered such that the input of a register only depends on the registers preceding it in the order.

4 Algebraic structures for signal flow graphs

The ring structure on streams forms the basis of the algebraic structures that characterize the behaviors of signal flow graphs. Recall that R is assumed to be a unital, commutative ring. In this section we describe the relevant algebras. We begin with the *ring $R[X]$ of polynomials over R*. It consists of all streams with only finitely many non-zero elements, i.e., streams of the form $[a_0] + [a_1] \times X + \cdots + [a_n] \times X^n$ for $a_0, \ldots, a_n \in R$. The following is a well-known fact about polynomials.

Proposition 6. *The set $R[X]$ of polynomials is a subring of R^ω.*

Polynomials are not closed under inverse, but we can extend the ring $R[X]$ to fractions of polynomials using a construction called *localization* [1]. Let U be the set of all invertible polynomial streams, i.e., $U = \{p \in R[X] \mid p(0) \in R^\times\}$.

We observe that U is multiplicatively closed, a necessary condition to form the *localization of $R[X]$ (viewed as an $R[X]$-module) at U*:

$$R[X]\left[U^{-1}\right] = \{[p : u] \mid p \in R[X], u \in U\} .$$

Elements in $R[X]\left[U^{-1}\right]$ are equivalence classes with respect to the relation $\sim$ on $R[X] \times U$ defined by

$$(p_1, u_1) \sim (p_2, u_2) \text{ iff } \exists v \in U : vp_1u_2 = vp_2u_1 .$$

Note that the extra v can be left out if, e.g., R is an integral domain. Using sum and convolution product of streams, we define addition and multiplication by scalars from $R[X]$ on $R[X]\left[U^{-1}\right]$ as follows:

$$\begin{aligned}[p_1 : u_1] + [p_2 : u_2] &= [p_1u_2 + p_2u_1 : u_1u_2] \\ q[p : u] &= [qp : u] .\end{aligned}$$

The above operations turn $R[X]\left[U^{-1}\right]$ into an $R[X]$-module with $[0 : 1]$ as additive identity, and such that, for all $u \in U$ and $p \in R[X]$, $u[p : uq] = [p : q]$ In fact, $R[X]\left[U^{-1}\right]$ is also a ring.

The behaviors of closed feedforward signal flow graphs are precisely the polynomial streams $R[X]$, whereas those of closed feedback signal flow graphs are the rational streams, i.e., the $R[X]$-module $R[X]\left[U^{-1}\right]$, cf. [23].

For open, feedforward signal flow graphs, we will show that the relevant algebraic structure is given by the free $R[X]$-module $R[X]^{\{\iota,1\}}$ generated by the two elements set $\{\iota, 1\}$. The intuition for these generators is that ι represents an unknown input stream, and 1 captures the initial content of the registers. Elements of $R[X]^{\{\iota,1\}}$ are of the form $p\iota + q1$ where $p, q \in R[X]$. In the sequel, we will write such an element simply as $p\iota + q$. As for closed signal flow graphs, allowing for feedback means constructing fractions. In the open case this means that we will consider the localization of $R[X]^{\{\iota,1\}}$ at the set U of invertible polynomials:

$$R[X]^{\{\iota,1\}}\left[U^{-1}\right] = \{[p\iota + q : u] \mid p, q \in R[X], u \in U\} .$$

Similar to the previous localization, fractions $[p\iota + q : u]$ are equivalence classes with respect to the relation $\sim$ on $R[X]^{\{\iota,1\}} \times U$ defined by:

$$(p_1\iota + q_1, u_1) \sim (p_2\iota + q_1, u_2) \text{ iff } \exists v \in U : v(p_1\iota + q)u_2 = v(p_2\iota + q)u_1 .$$

As usual, we write $\frac{p\iota+q}{u}$ instead of $[p\iota + q : u]$. Addition and scalar multiplication are defined as expected, turning $R[X]^{\{\iota,1\}}\left[U^{-1}\right]$ into an $R[X]$-module which is free among all $R[X]$-modules M for which the assignment $\lambda_u : x \mapsto ux$ is a linear isomorphism on M for all $u \in U$.

Proposition 7. *The $R[X]$-module $R[X]^{\{\iota,1\}}\left[U^{-1}\right]$ together with the linear inclusion map $\varphi \colon R[X]^{\{\iota,1\}} \to R[X]^{\{\iota,1\}}\left[U^{-1}\right]$ given by $\varphi(x) = \frac{x}{1}$ satisfies the following universal property [1,8].*

If M is an $R[X]$-module such that for all $u \in U$, the linear maps $\lambda_u \colon M \to M$ given by $\lambda_u(x) = ux$ are isomorphisms, and if $f \colon R[X]^{\{\iota,1\}} \to M$ is a linear map, then there is a unique linear map $\overline{f} \colon R[X]^{\{\iota,1\}}\left[U^{-1}\right] \to M$ extending f such that the following diagram commutes:

$$\begin{tikzcd}
R[X] \arrow[r, "\varphi"] \arrow[dr, "f"'] & R[X]^{\{\iota,1\}}\left[U^{-1}\right] \arrow[d, dashed, "!\overline{f}"] \\
 & M
\end{tikzcd}$$

The extension of f is given by $\overline{f}([x : u]) = \lambda_u^{-1}(f(x))$. Moreover, if f is injective, then its extension $\overline{f}$ is injective as well.

The following lemma relates the algebraic constructions used so far.

Lemma 8. *There are inclusions among the constructed $R[X]$-modules, as indicated in the following commuting diagram*

$$\begin{tikzcd}
R[X]\left[U^{-1}\right] \arrow[r, hook, "j"] & R[X]^{\{\iota,1\}}\left[U^{-1}\right] \\
R[X] \arrow[u, hook, "\varphi_1"] \arrow[r, hook, "i"] & R[X]^{\{\iota,1\}} \arrow[u, hook, "\varphi_2"]
\end{tikzcd}$$

where φ_1, φ_2 are the inclusions into the localizations, i is given by $x \mapsto x1$, and j is the linear extension $j = \overline{\varphi_2 \circ i}$. All these inclusions are injective $R[X]$-module homomorphisms.

The elements of the four algebras above denote causal stream functions. The polynomials $R[X]$ are by definition streams, or, equivalently, constant stream transformations (which are clearly causal). For the algebras corresponding to feedforward, closed and arbitrary circuits we have semantic maps $[\![-]\!]_{\mathrm{ff}}, [\![-]\!]_{\mathrm{c}}$ and $[\![-]\!]$ as shown here in the following diagram (we give their definitions below):

$$\begin{tikzcd}
R[X]^{\{\iota,1\}} \arrow[r, hook] \arrow[dr, "{[\![-]\!]_{\mathrm{ff}}}"'] & R[X]^{\{\iota,1\}}\left[U^{-1}\right] \arrow[d, "{[\![-]\!]}"] & R[X]\left[U^{-1}\right] \arrow[l, hook'] \arrow[dl, "{[\![-]\!]_{\mathrm{c}}}"] \\
\{\iota,1\} \arrow[u, hook] \arrow[r, "g"'] & C(R^\omega) &
\end{tikzcd}$$

Here $g \colon \{\iota,1\} \to C(R^\omega)$ is the map defined by $g(\iota) = \mathrm{id}_{R^\omega}$ and $g(1) = \sigma \mapsto [1]$, for all $\sigma \in R^\omega$. Since $R[X]^{\{\iota,1\}}$ is the *free* $R[X]$*-module* over the set $\{\iota,1\}$ the map $[\![-]\!]_{\mathrm{ff}} \colon R[X]^{\{\iota,1\}} \to C(R^\omega)$ is defined as the unique linear map extending g.

For all $u \in U$, the scalar multiplication $\lambda_u \colon C(R^\omega) \to C(R^\omega)$ on $C(R^\omega)$, sends f to $u \times f$ (point-wise convolution). Since U consists of all invertible polynomial streams, λ_u has an inverse $\lambda_u^{-1}(f) = u^{-1} \times f$, and hence each

λ_u is a linear isomorphism on $C(R^\omega)$. We can thus apply the universal property of the localization (Prop. 7) in order to uniquely define the linear map $[\![-]\!]\colon R[X]^{\{\iota,1\}}\left[U^{-1}\right] \to C(R^\omega)$ as the extension of $[\![-]\!]_{\mathrm{ff}}\colon R[X]^{\{\iota,1\}} \to C(R^\omega)$.

Finally, the map $[\![-]\!]_{\mathrm{c}}\colon R[X]\left[U^{-1}\right] \to C(R^\omega)$ is obtained by restricting $[\![-]\!]$ along the inclusion $R[X]\left[U^{-1}\right] \hookrightarrow R[X]^{\{\iota,1\}}\left[U^{-1}\right]$. Note that $[\![[p:q]]\!]_{\mathrm{c}}$ is simply the constant map that sends every stream to the rational stream $p \times q^{-1}$.

Theorem 9 (Soundness and completeness). *For all $x, y \in R[X]^{\{\iota,1\}}\left[U^{-1}\right]$, $x = y$ if and only if $[\![x]\!] = [\![y]\!]$.*

Soundness is, in fact, what allows us to define $[\![-]\!]\colon R[X]^{\{\iota,1\}}\left[U^{-1}\right] \to C(R^\omega)$ as (linear) map. Completeness, on the other hand, is a consequence of the fact that all the maps $[\![-]\!]_{\mathrm{ff}}$, $[\![-]\!]_{\mathrm{c}}$, and $[\![-]\!]$ are injective, since id_{R^ω} and $\sigma \mapsto [1]$ are linearly independent in the $R[X]$-module $C(R^\omega)$.

5 Solving linear systems, algebraically

In this section we give a matrix-based method for computing the solution of an *open* linear system. Our method is a novel adaptation of the method presented in [23] for solving *closed* linear systems.

Let $L = (V, M, O, I)$ be an n-dimensional open linear system with (stream) variables $V = \{v_1, \ldots, v_n\}$. We use the fact that the matrix M over R can be seen as a matrix over $R[X]$ by applying the inclusion $a \mapsto [a]$ to each entry in M. This allows us to multiply M (entry-wise) with scalars from $R[X]$. Likewise, we implicitly apply the entry-wise inclusion to view I as a vector over $R[X]$. More abstractly, we are using the $R[X]$-algebra structure on the matrices.

Informally, a state solution to L is an assignment $s_\sigma\colon V \to R^\omega$, which depends on an input stream σ, and satisfies the equations of L when taking $\iota = \sigma$. By the fundamental theorem of stream calculus (Prop. 2), for any such assignment, we have $s_\sigma(v_i) = [s_\sigma(v_i)(0)] + s_\sigma(v_i)'X$ for each $v_i \in V$. Hence, as a system of constraints L is equivalent to the system expressed by:

$$s_\sigma = I + (MX)\begin{pmatrix}\iota\\ s_\sigma\end{pmatrix}$$

This system is, in turn, equivalent to the "square" system:

$$\begin{pmatrix}\iota\\ s_\sigma\end{pmatrix} = \begin{pmatrix}\iota\\ I\end{pmatrix} + \begin{pmatrix}\mathbf{0}\\ MX\end{pmatrix}\begin{pmatrix}\iota\\ s_\sigma\end{pmatrix}$$

where $\mathbf{0}$ denotes a row of $n+1$ 0's. Finally, this system is equivalent to

$$\left(\mathbf{I}_{n+1} - \begin{pmatrix}\mathbf{0}\\ MX\end{pmatrix}\right)\begin{pmatrix}\iota\\ s_\sigma\end{pmatrix} = \begin{pmatrix}\iota\\ I\end{pmatrix} \tag{2}$$

where $\mathbf{I}_{n+1}$ is the $(n+1)$-dimensional identity matrix. Formally, the system (2) can be seen as an equation over the $\overline{V}$-fold direct sum (copower) $\overline{V} \odot \Gamma = \bigoplus_{\overline{V}} \Gamma$

of Γ where $\Gamma = R[X]^{\{\iota,1\}}\left[U^{-1}\right]$, as before. Elements of $\overline{V}\odot\Gamma$ are $(n+1)$-vectors, and to ease readability, we introduce the notation $\left(\begin{smallmatrix}\iota\\ x\end{smallmatrix}\right) = \iota v_{\text{in}} + x \in \overline{V}\odot\Gamma$ where $x = (x_1, \ldots, x_n) \in V\odot\Gamma$. Using the usual matrix-vector product, we can apply M to vectors from the $R[X]$-module $\overline{V}\odot\Gamma$, and solve the system using standard methods from linear algebra [1]. Note also that (2) does not depend on the output O of the linear system. We therefore call it a *state equation*, and a solution will be referred to as a *state solution* of L.

Definition 10. *Let $L = (V, M, O, I)$ be a linear system. A* state solution to L *is an element $s \in V\odot\Gamma$ such that*

$$\left(\mathbf{I}_{n+1} - \begin{pmatrix}\mathbf{0}\\ MX\end{pmatrix}\right)\begin{pmatrix}\iota\\ s\end{pmatrix} = \begin{pmatrix}\iota\\ I\end{pmatrix}.$$

A solution s to L thus contains for each $v_i \in V$ an element s_i in the localization $\Gamma = R[X]^{\{\iota,1\}}\left[U^{-1}\right]$. Since we have seen in the previous section that localization elements denote causal functions, we get a stream assignment $s_\sigma\colon V \to R^\omega$ defined by $s_\sigma(v_i) = [\![s_i]\!](\sigma)$, for all streams $\sigma \in R^\omega$. The dependency on the input ι is thus formalized via the free module $R[X]^{\{\iota,1\}}$.

Proposition 11. *Every n-dimensional open linear system $L = (V, M, O, I)$ has a unique state solution $s \in V\odot\left(R[X]^{\{\iota,1\}}\left[U^{-1}\right]\right)$.*

If $s \in V\odot\Gamma$ is the unique state solution of L, then the *output solution of* L is the localization element obtained by applying the output map O to the $(n+1)$-vector $\iota v_{\text{in}} + s$, that is, the output solution of L is defined as

$$O(\iota v_{\text{in}} + s) \in R[X]^{\{\iota,1\}}\left[U^{-1}\right].$$

Note that the output solution is uniquely defined for L due to the existence of a unique state solution.

Definition 12. *If L is an open linear system, then we define the* algebraic solution to L, *denoted by $[\![L]\!]$, as the algebraic semantics of the output solution: $[\![L]\!] = [\![O(\iota v_{\text{in}} + s)]\!] \in C(R^\omega)$.*

In Section 7 we will show that $[\![L]\!]$ is indeed the behavior of any signal flow graph represented by L.

Example 13. We solve the system from Example 3 using the above method.

$$\begin{pmatrix}1 & 0 & 0\\ -X & 1 & 0\\ -X & X & 1-X\end{pmatrix}\begin{pmatrix}\iota\\ s_1\\ s_2\end{pmatrix} = \begin{pmatrix}\iota\\ 0\\ 0\end{pmatrix}$$

Using Gaussian elimination, we find the state solutions $s_1 = \iota X$, $s_2 = \frac{\iota X - \iota X^2}{1-X} = \iota X$ and the output solution

$$o = 1\iota - \iota X + 1\frac{\iota X - \iota X^2}{1-X} = \iota - \iota X + \iota X = \iota.$$

Hence the composition $\int(\Delta\iota)$ is the identity, as expected.

6 Solving linear systems, coalgebraically

In this section we describe a coalgebraic method of associating a stream transformation to a linear system. The key observation is that each linear system can be viewed as a Mealy machine. The *Mealy machines* we will consider are coalgebras for the functor $\mathcal{M} = (R \times (-))^R$. Intuitively, for a set of states S, a structure map $c\colon S \to \mathcal{M}S$ assigns to each state $x \in S$ and input value $a \in R$ a pair $(o, y) \in R \times S$, where o is the output and y is the next state of the machine in state x on input a.

A final $\mathcal{M}$-coalgebra exists, and is given by the set $C(R^\omega)$ of causal stream transformations equipped with the structure map $\delta\colon C(R^\omega) \to (R \times C(R^\omega))^R$ defined for all $f \in C(R^\omega)$ and $a \in R$ by $\delta(f)(a) = (f[a], f^{(a)})$ where for all $\sigma \in R^\omega$,

$$\begin{aligned} f[a] &= f(a:\sigma)(0) && \in R, \\ f^{(a)}(\sigma) &= f(a:\sigma)' && \in R^\omega, \end{aligned} \tag{3}$$

(see e.g. [9]). Note that because f is causal the definition of $f[a]$ is independent of the choice of $\sigma \in R^\omega$, and $f^{(a)}$ is causal as well.

By instantiating the notion of bisimulation to the functor $\mathcal{M}$, we obtain that a bisimulation between Mealy machines (S_1, c_1) and (S_2, c_2) is a relation $B \subseteq S_1 \times S_2$ such that for all $(s_1, s_2) \in B$ and all $a \in R$ the following holds: if $(o_1, t_1) = c_1(s_1)(a)$ and $(o_2, t_2) = c_2(s_2)(a)$ then

$$o_1 = o_2 \qquad \text{and} \qquad (t_1, t_2) \in B.$$

Definition 14 (Linear machines). *Let $L = (V, M, O, I)$ be an open linear system with extra input variable v_{in}. We define the* Mealy machine associated with L *as the Mealy machine (R^V, c_L), where $c_L\colon R^V \to (R \times R^V)^R$ is defined by*

$$c_L(x)(a) = (O(av_{\text{in}} + x), M(av_{\text{in}} + x))$$

for all $a \in R$ and $x \in R^V$.

We point out that for the linear machine (R^V, c_L) associated with some L, the actual structure map $c_L\colon R^V \to (R \times R^V)^R$ is, in general, not linear, only its uncurried form $\overline{c_L}\colon R^V \times R \to R \times R^V$ is linear.

A Mealy machine (S, c), is called a *linear machine* if S is an R-module and the uncurried structure map $\overline{c}\colon S \times R \to R \times S$ is linear. Clearly, not all Mealy machines are linear machines. In particular, the final $\mathcal{M}$-coalgebra $(C(R^\omega), \delta)$ is not a linear machine, because $\overline{\delta}$ is *not* linear, even though δ itself is linear. So the final Mealy machine is linear in the (coalgebraic) "curried form" whereas linear machines are linear in "uncurried form".

We denote by $\langle\text{LM}\rangle$ the least subcoalgebra of $(C(R^\omega), \delta)$ containing the behaviors of all linear machines. Similarly, $\langle\text{LM}_\text{c}\rangle$ and $\langle\text{LM}_\text{ff}\rangle$ denote the least subcoalgebras of $(C(R^\omega), \delta)$ containing the behaviors of all linear machines associated to closed, respectively feedforward, circuits.

We define the coalgebraic solution of an open linear system via the final Mealy machine.

Definition 15. *Let $L = (V, M, O, I)$ be an open linear system. We define the* coalgebraic solution *of L, denoted by $\llparenthesis L \rrparenthesis$, to be the coalgebraic behavior of the initial state I in its Mealy machine (R^V, c_L), that is, $\llparenthesis L \rrparenthesis = \widetilde{c_L}(I) \in C(R^\omega)$.*

Example 16. Taking the linear system in Example 3 (our running example), we calculate the first few outputs and states of the resulting Mealy machine with input stream $\sigma \in R^\omega$. To this end, we set $s_0 = I$ and $(o_k, s_{k+1}) = c_L(s_k)(\sigma_k)$ and compute the concrete values:

$$\begin{aligned}
o_0 &= O(\sigma_0 v_{\text{in}} + s_0) = \sigma_0 - 0 + 0 = \sigma_0 \\
s_1 &= M(\sigma_0 v_{\text{in}} + s_0) = \sigma_0 v_1 + (\sigma_0 - 0 + 0) v_2 = \sigma_0 v_1 + \sigma_0 v_2 \\
o_1 &= O(\sigma_1 v_{\text{in}} + s_1) = \sigma_1 - \sigma_0 + \sigma_0 = \sigma_1 \\
s_2 &= M(\sigma_1 v_{\text{in}} + s_1) = \sigma_1 v_1 + (\sigma_1 - \sigma_0 + \sigma_0) v_2 = \sigma_1 v_1 + \sigma_1 v_2 \\
&\vdots
\end{aligned}$$

Clearly, we get $\llparenthesis L \rrparenthesis(\sigma) = \widetilde{c_L}(I)(\sigma) = (o_0, o_1, \dots) = \sigma$ as expected.

7 Algebraic and coalgebraic solutions coincide

In the previous two sections we have seen an algebraic and a coalgebraic method for assigning a causal stream transformation to a linear system. In this section we will show that the two methods lead to the same element of $C(R^\omega)$.

To begin with, we show that the localization $R[X]^{\{\iota,1\}}\left[U^{-1}\right]$ can be given a Mealy machine structure such that the algebraic semantics $[\![-]\!]$ coincides with its coalgebraic behavior map. For a more compact notation, we define $\mathit{\Gamma} = R[X]^{\{\iota,1\}}\left[U^{-1}\right]$. The Mealy structure on $\mathit{\Gamma}$ is defined by a two-step procedure that mimics the definition of the structure map of the final Mealy machine in (3): For all causal functions $f \in C(R^\omega)$,

$$f \longmapsto \lambda a \in R. f(a : -) \longmapsto \lambda a \in R. \lambda \sigma \in R^\omega. (f(a : \sigma)(0), f(a : \sigma)') \,.$$

Let $x \in \mathit{\Gamma}$. To mimic the leftmost step above, we need an element $x_a \in \mathit{\Gamma}$ such that $[\![x_a]\!] = [\![x]\!](a{:}-)$. The idea is to obtain x_a by substituting $a{:}\iota = a + \iota X$ for ι in x. Formally, we define the *substitution* $x[y/\iota]$ of $y \in R[X]^{\{\iota,1\}}$ for ι in x as the linear extension of the map $\rho_y \colon \{\iota, 1\} \to \mathit{\Gamma}$ with $\rho_y(\iota) = y$ and $\rho_y(1) = 1$, i.e., $x[y/\iota] \coloneqq \overline{\rho_y}(x)$. More concretely, for $x = (p\iota + q)u^{-1}$ and $y = r\iota + t$, we have

$$x[y/\iota] = (pr\iota + (pt + q))u^{-1} \,.$$

Lemma 17. *For all $x \in R[X]^{\{\iota,1\}}\left[U^{-1}\right]$, $a \in R$ and $\sigma \in R^\omega$,*

$$[\![x[(a + \iota X)/\iota]]\!](\sigma) = [\![x]\!](a : \sigma) \,.$$

We now make the observation, that $x[(a+\iota X)/\iota]$ lies in the submodule $G \subseteq \Gamma$ where ι always occurs "guarded", namely in the form ιX:

$$G = \left\{ \frac{p\iota X + q}{u} \;\middle|\; p, q \in R[X], u \in U \right\}.$$

Due to the guardedness, we can define linear maps $o\colon G \to R$ and $d\colon G \to \Gamma$, which should be thought of as taking initial value and derivative of guarded stream expressions, by inductively applying the behavioral differential equations from Section 2.3 with the special case that $o(\iota X) = 0$ and $d(\iota X) = \iota$. The Mealy machine on Γ is obtained by composing the substitution with the maps $o\colon G \to R$ and $d\colon G \to \Gamma$.

Definition 18. *The localization $\Gamma = R[X]^{\{\iota,1\}}\left[U^{-1}\right]$ can be turned into a Mealy machine with the structure map $\gamma\colon \Gamma \to (R \times \Gamma)^R$ defined by*

$$\gamma(x)(a) = (o(x[(a+\iota X)/\iota]), d(x[(a+\iota X)/\iota]))\,.$$

Concretely, for $x = \frac{p\iota+q}{u} \in \Gamma$ and $a \in R$ we have:

$$\begin{aligned} o(x[(a+\iota X)/\iota]) &= \frac{p(0)a + q(0)}{u(0)} \\ d(x[(a+\iota X)/\iota]) &= \frac{p}{u}\iota + \frac{u(0)(p'a+q') - (p(0)a+q(0))u'}{u(0)u}\,. \end{aligned}$$

Since we defined o and d inductively using the behavioral differential equations, one can show using Lem. 17 and Def. 18 that for all $x \in \Gamma$, all $a \in R$ and all $\sigma \in R^\omega$:

$$\begin{aligned} o(x[(a+\iota X)/\iota]) &= [\![x]\!](a:\sigma)(0) \\ [\![d(x[(a+\iota X)/\iota])]\!](\sigma) &= [\![x]\!](a:\sigma)'\,. \end{aligned}$$

In other words, we have shown the following lemma.

Lemma 19. *The algebraic semantics $[\![-]\!]\colon \Gamma \to C(R^\omega)$ is a Mealy machine homomorphism from (Γ, γ) to $(C(R^\omega), \delta)$. By finality of $(C(R^\omega), \delta)$, the algebraic semantics coincides with the coalgebraic behavior map, that is, $[\![x]\!] = \widetilde{\gamma}(x)$ for all $x \in \Gamma$.*

We will use the above lemma to show our main result.

Theorem 20. *For any open linear system L, the algebraic solution of L coincides with the coalgebraic solution of L: $[\![L]\!] = \langle\!|L|\!\rangle$.*

Proof. Let $L = (V, M, O, I)$ be an n-dimensional open linear system and let $s \in V \odot \Gamma$ be the unique state solution of L. Furthermore, let (R^V, c_L) be the Mealy machine associated to L with initial state I. The proof is divided into two steps. We leave out some details.

First, we construct a Mealy machine $(V \odot \Gamma, d\colon V \odot \Gamma \to \mathcal{M}(V \odot \Gamma))$ on the solution space. The map d is obtained by applying the Mealy structure (Γ, γ)

point-wise. For $x = (x_1, ..., x_n) \in V \odot \Gamma$ and $a \in R$, let $d(x)(a) = (O(av_{\text{in}} + o), x')$ where $o = \sum_{i=1}^{n} o_i v_i$ and $x' = \sum_{i=1}^{n} x_i' v_i$ with $\gamma(x_i)(a) = (o_i, x_i')$ for $i = 1, \ldots, n$. With this definition, we can show that for all $x \in V \odot \Gamma$, $\widetilde{d}(x) = \widetilde{\gamma}(O(\iota v_{\text{in}} + x))$. By Lem. 19 we get $[\![O(\iota v_{\text{in}} + x)]\!] = \widetilde{\gamma}(O(\iota v_{\text{in}} + x))$, and hence $[\![O(\iota v_{\text{in}} + x)]\!] = \widetilde{d}(x)$.

In the second step we build a bisimulation B between $(V \odot \Gamma, d)$ and (R^V, c_L) with $(s, I) \in B$ from which $\widetilde{d}(s) = \widetilde{c_L}(I)$ and hence $[\![O(\iota v_{\text{in}} + s)]\!] = \widetilde{c_L}(I)$ follows, as desired. The relation B is constructed in the following way. We denote by $I_{\sigma[n]} \in R^V$ the state reached in (R^V, c_L) after reading $\sigma[n] = (\sigma(0), \ldots, \sigma(n-1))$ when starting in state I. For 0 we put $I_{\sigma[0]} = I$ for all $\sigma \in R^\omega$. Analogously, we denote by $s_{\sigma[n]} \in V \odot \Gamma$ the state reached in the Mealy machine $(V \odot \Gamma, d)$ after reading $\sigma[n]$ when starting from the state solution s, and again we put $s_{\sigma[0]} = s$.

Using the fact that s is a state solution, we can show that $s_{\sigma[n]}$ is obtained by applying M repeatedly to the inputs $\sigma(0), \ldots, \sigma(n-1)$ and s. To this end, let $\pi \colon \overline{V} \odot \Gamma \to V \odot \Gamma$ be the evident projection and $M^{[k]} = \pi \circ \left(\begin{smallmatrix}\mathbf{0}\\ M\end{smallmatrix}\right)^k$ the k-fold composition of M, followed by this projection. Furthermore, we define $x_n = \sum_{i=0}^{n-1} \sigma_i X^i + \iota X^n \in \Gamma$ and observe, that for $n \geq 1$ the x_n is in fact an element of G, the submodule of Γ in which ι occurs in guarded form. With a bit of patience one arrives at the following explicit definitions

$$\begin{aligned}
s_{\sigma[n]} &= M^{[n]} \begin{pmatrix} 0 \\ s[x_n/\iota] \end{pmatrix} + \sum_{k=0}^{n-1} M^{[k+1]} \begin{pmatrix} x_n^{(n-k-1)} \\ \mathbf{0} \end{pmatrix} \\
I_{\sigma[n]} &= M^{[n]} \begin{pmatrix} 0 \\ I \end{pmatrix} \qquad\quad + \sum_{k=0}^{n-1} M^{[k+1]} \begin{pmatrix} \sigma_{n-k-1} \\ \mathbf{0} \end{pmatrix}.
\end{aligned}$$

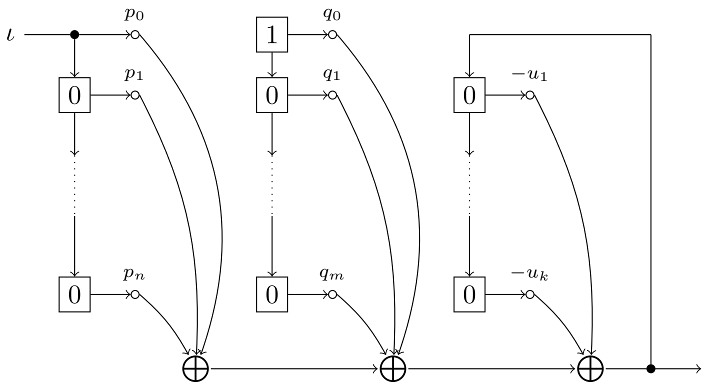
It is now easy to see that, applying o, we have $o\Big(x_n^{(n-k-1)}\Big) = \sigma_{n-k-1}$ and $o(s[x_n/\iota]) = I$. By point-wise application of o we deduce, that the outputs of $d\big(s_{\sigma[n]}\big)(a)$ and $c_L\big(I_{\sigma[n]}\big)(a)$ match for all $a \in R$. Moreover, one can also show, that the next states are $s_{\tau[n+1]}$ and $I_{\tau[n+1]}$ for some $\tau \in R^\omega$ with $\tau[n] = \sigma[n]$ and $\tau(n) = a$. Thus the relation $B = \big\{\big(s_{\sigma[n]}, I_{\sigma[n]}\big) \bigm| \sigma \in R^\omega, n \in \mathbb{N}\big\}$ is a bisimulation. Finally, by definition $(s, I) = \big(s_{\sigma[0]}, I_{\sigma[0]}\big) \in B$ and the claim follows. □

We end this section with a Kleene-style theorem showing that the module $R[X]^{\{\iota,1\}}\left[U^{-1}\right]$ characterizes precisely the behaviors of *all* open linear systems.

Theorem 21. *For every open linear system L, the unique output solution x of L is in $R[X]^{\{\iota,1\}}\left[U^{-1}\right]$. Conversely, for every $x \in R[X]^{\{\iota,1\}}\left[U^{-1}\right]$ there is an open linear system L such that x is the output solution of L.*

Proof. One direction of the above theorem is just Prop. 11. In order to prove the other direction, we sketch here how to construct from $x \in R[X]^{\{\iota,1\}}\left[U^{-1}\right]$ an open linear system L with x as output solution. Assume $x = \frac{p\iota + q}{u}$ where, without loss of generality, $u_0 = 1$.

We take as L the linear system associated with the following signal flow graph:



The register contents are represented as a state vector $s = \begin{pmatrix} s_p & s_q & s_u \end{pmatrix}^T \in R^{n+(m+1)+k}$ separated into $s_p \in R^n$, $s_q \in R^{m+1}$ and $s_u \in R^k$.

More concretely, $L = (V, M, O, I)$ with variables $V = V_p + V_q + V_u$ using $V_p = \{1, \ldots, n\}, V_q = \{1, \ldots, m+1\}$ and $V_u = \{1, \ldots, k\}$ is defined as follows.

The initial state is $I = (I_p, I_q, I_u)$ where

$$I_p = (0, \ldots, 0) \qquad I_q = (1, 0, \ldots, 0) \qquad I_u = (0, \ldots, 0).$$

We describe the matrix M by letting $s^{(a)} = M(av_{\text{in}} + s) = M(av_{\text{in}} + s_p + s_q + s_u)$:

$$\begin{aligned} s^{(a)}_{p,1} &= a & s^{(a)}_{p,i+1} &= s_{p,i}, \quad 1 \le i \le n-1 \\ s^{(a)}_{q,1} &= 0 & s^{(a)}_{q,i+1} &= s_{q,i}, \quad 1 \le i \le m \\ s^{(a)}_{u,1} &= O(av_{\text{in}} + s) & s^{(a)}_{u,i+1} &= s_{u,i}, \quad 1 \le i \le k-1. \end{aligned}$$

The output matrix O is

$$O = (p_0, p_1, \ldots, p_n, q_0, \ldots, q_m, -u_1, \ldots, -u_k).$$

To prove that the output behavior of this linear system is x we consider the Mealy machine $(R^{n+(m+1)+k}, c_L)$ associated to L. Consider now the relation $B \subseteq R^{n+(m+1)+k} \times \Gamma$ given by

$$B = \left\{ (s, z) \;\middle|\; z = \frac{p}{u}\iota + \frac{1}{u}\left(\sum_{i=1}^{n} s_{p,i} p^{(i)} + \sum_{i=1}^{m+1} s_{q,i} q^{(i-1)} - \sum_{i=1}^{k} s_{u,i} u^{(i)} \right) \right\}$$

One can verify that B is a bisimulation between $(R^{n+(m+1)+k}, c_L)$ and (Γ, γ), and that $(I, x) \in B$, hence we have that $\widetilde{c_L}(I) = \widetilde{\gamma}(x) = [\![x]\!]$.

By solving the system L algebraically, we get a unique output solution $x_o \in R[X]^{\{\iota,1\}}\left[U^{-1}\right]$ with $[\![x_o]\!] = \widetilde{c_L}(I)$ by Thm. 20. Since $[\![-]\!]$ is injective (Thm. 9) we have $x = x_o$. $\square$

Due to Thm. 21, given an open linear system L, we can simply refer to $[\![L]\!] = \langle\!| L |\!\rangle$ as the *behavior of* L. Given a signal flow graph C, we refer to $[\![\mathcal{L}(C)]\!] = \langle\!| \mathcal{L}(C) |\!\rangle$ as the *behavior of* C.

We can use Thm. 21 to give a precise characterization of the behaviors of the subclasses of closed, respectively feedforward, circuits.

Corollary 22. *We have the following axiomatizations:*

a) The behaviors $\langle \mathrm{LM} \rangle$ *of all circuits is an* $R[X]$*-submodule of* $C(R^\omega)$*, and* $\langle \mathrm{LM} \rangle \cong R[X]^{\{\iota,1\}}\left[U^{-1}\right]$ *as* $R[X]$*-modules and as Mealy machines.*
b) The behaviors $\langle \mathrm{LM_c} \rangle$ *of closed, feedback circuits is an* $R[X]$*-submodule of* $C(R^\omega)$*, and* $\langle \mathrm{LM_c} \rangle \cong R[X]\left[U^{-1}\right]$ *as* $R[X]$*-modules and as Mealy machines.*
c) The behaviors $\langle \mathrm{LM_{ff}} \rangle$ *of open, feedforward circuits is an* $R[X]$*-submodule of* $C(R^\omega)$*, and* $\langle \mathrm{LM_{ff}} \rangle \cong R[X]^{\{\iota,1\}}$ *as* $R[X]$*-modules and as Mealy machines.*

Proof. **a)** It follows immediately from Thm. 21 that the image of the algebraic semantics map $[\![-]\!] \colon R[X]^{\{\iota,1\}}\left[U^{-1}\right] \to C(R^\omega)$ is $\langle \mathrm{LM} \rangle$, and by soundness and completeness (Thm. 9), $[\![-]\!]$ is an injective linear map of $R[X]$-modules, hence a module isomorphism from $R[X]^{\{\iota,1\}}\left[U^{-1}\right]$ to $\langle \mathrm{LM} \rangle$. From the fact that $[\![-]\!] \colon R[X]^{\{\iota,1\}}\left[U^{-1}\right] \to \langle \mathrm{LM} \rangle$ is also a bijective Mealy homomorphism, it follows that $R[X]^{\{\iota,1\}}\left[U^{-1}\right]$ and $\langle \mathrm{LM} \rangle$ are also isomorphic as Mealy machines.
Since $[\![-]\!]$ is both an $R[X]$-linear map and a Mealy homomorphism which moreover is injective, it suffices for the remaining items to show that the restrictions of $[\![-]\!]$ to $R[X]\left[U^{-1}\right]$ and $R[X]^{\{\iota,1\}}$ have range $\langle \mathrm{LM_c} \rangle$ and $\langle \mathrm{LM_{ff}} \rangle$, respectively.

b) Let $x = \frac{q}{u} \in R[X]\left[U^{-1}\right]$ and $j(x) = \frac{0\iota+q}{u} \in R[X]^{\{\iota,1\}}\left[U^{-1}\right]$ its embedding. Then we construct a circuit C for x such that $[\![x]\!]_{\mathrm{c}} = [\![j(x)]\!] = [\![\mathcal{L}(C)]\!]$, following the proof of Thm. 21. By construction, this circuit will be independent of the input, i.e., C is closed and hence $[\![x]\!]_{\mathrm{c}} \in \langle \mathrm{LM_c} \rangle$. Conversely, if C is a closed, feedback circuit and $L = (V, M, O, I)$ its associated linear system, then the first column of O is 0 (cf. Lem. 5). Consequently, the output solution of L is of the form $j(x) = \frac{0\iota+q}{u}$ for some $x \in R[X]\left[U^{-1}\right]$ with $\langle\!| L |\!\rangle = [\![j(x)]\!] = [\![x]\!]_{\mathrm{c}}$, thus $[\![-]\!]_{\mathrm{c}} \colon R[X]\left[U^{-1}\right] \to \langle \mathrm{LM_c} \rangle$ is onto.

c) Let $x = p\iota + q \in R[X]^{\{\iota,1\}}$ and $\varphi_2(x) = \frac{p\iota+q}{1} \in R[X]^{\{\iota,1\}}\left[U^{-1}\right]$ its embedding. Then we construct again a circuit C for x such that $[\![x]\!]_{\mathrm{ff}} = [\![\varphi_2(x)]\!] = [\![\mathcal{L}(C)]\!]$, following the proof of Thm. 21. By construction, C is feedforward (since $u_1, \ldots, u_k$ will all be 0), and hence $[\![x]\!]_{\mathrm{ff}} \in \langle \mathrm{LM_{ff}} \rangle$. Conversely, if C is an open, feedforward circuit and $L = (V, M, O, I)$ its associated linear system, then M is of the "lower-triangular form" given in Lem. 5. It follows that the output solution of L will be of the form $\varphi(x) = \frac{p\iota+q}{1}$ for some $x \in R[X]^{\{\iota,1\}}$ with $\langle\!| L |\!\rangle = [\![\varphi(x)]\!] = [\![x]\!]_{\mathrm{ff}}$, so $[\![-]\!]_{\mathrm{ff}} \colon R[X]\left[U^{-1}\right] \to \langle \mathrm{LM_{ff}} \rangle$ is onto. $\square$

Remark 23. From the coalgebraic point of view, one may wonder, whether any of our algebraic characterizations of open signal flow graphs is a fixed point for the functor $\mathcal{M}$ of Mealy machines. For *closed* feedback signal flow graphs, the localization $R[X]\left[U^{-1}\right]$ is a fixed point of the functor for streams [4,19]. In the general case of *open* signal flow graphs the result is negative, i.e., $\gamma\colon \mathit{\Gamma} \to (R \times \mathit{\Gamma})^R$ is *not* an isomorphism, for $\mathit{\Gamma} = R[X]^{\{\iota,1\}}\left[U^{-1}\right]$. It is indeed easy to see that d is not surjective by taking, for example $f \in (R\times\mathit{\Gamma})^R$ with $f(a) = (1,\iota) \in R\times\mathit{\Gamma}$ for all $a \in R$. Let us assume there is an element $x = \frac{p\iota+q}{u} \in \mathit{\Gamma}$ with $d(x) = f$. Then we necessarily have $h(x[(a + \iota X)/\iota]) = 1$ and we can deduce that $p_0 = 0$, for otherwise a cannot be arbitrary. It follows that x is of the following form (for new $p, q, u \in R[X]$)

$$x = \frac{b + qX + pX\iota}{b + uX}$$

From the requirement that $(x[(a+\iota X)/\iota])' = \iota$, we can derive that $paX\iota = b+uX$ and hence, by taking initial output, that $0 = b$ which is a contradiction. Thus there is no $x \in \mathit{\Gamma}$ with $d(x) = f$, i.e., d is not surjective and therefore $\mathit{\Gamma}$ is not a fixed point of the functor $\mathcal{M}$.

8 Concluding remarks

Our main contribution in this paper is the axiomatization of signal flow graphs using standard mathematical concepts and techniques, such as polynomials and module localization. In the following table we give an overview of the algebras corresponding to different classes of signal flow graphs.

type	feedforward	feedback
closed	Free R-algebra $R[X]$ of polynomials	Localization $R[X]\left[U^{-1}\right]$
open	Free $R[X]$-module $R[X]^{\{\iota,1\}}$	Localization $R[X]^{\{\iota,1\}}\left[U^{-1}\right]$

Our results yield a method for deciding circuit equivalence by comparing solutions in the localization $\mathit{\Gamma} = R[X]^{\{\iota,1\}}\left[U^{-1}\right]$. Deciding whether $\frac{p_1\iota+q_1}{u_1} = \frac{p_2\iota+q_2}{u_2}$ boils down to finding a $v \in U$ such that $v(p_1u_2-p_2u_1) = 0$ and $v(q_1u_2-q_2p_1) = 0$ hold (using that $R[X]^{\{\iota,1\}}$ is freely generated). If R is an integral domain, then this problem reduces to the simple problem of deciding equivalence of polynomials: $p_1u_2 = p_2u_1$ and $q_1u_2 = q_2p_1$. If equality in R is effectively decidable, then polynomial equivalence is effectively decidable, since $R[X]$ is the free commutative R-algebra over the single generator X. Summarizing, if R is an integral domain in which equality is effectively decidable, then so is equality in $\mathit{\Gamma}$.

We have restricted our attention to circuits with at most one input end, and one output end. It is straightforward to extend our result to more inputs by using different generators $\iota_1, \ldots, \iota_k$ for each input end. Multiple outputs, on the other hand, can be represented by changing the underlying ring to R^m (with component-wise operations).

All the work in this paper is based on the assumption that signals are elements of a *commutative* ring. There are, however, interesting rings used in systems theory which are non-commutative, such as the ring of matrix polynomials [27]. An interesting future direction is the generalization of our results using non-commutative localization. This raises two problems: first one needs different conditions on the ring to still have an invariant basis number, so that matrices still represent linear maps. The second problem is that in the localization one generally loses the property that every element is of the form $\frac{a}{u}$, instead they will be sums of such fractions. For discussions on these issues see for example [14].

The localization Γ and the causal functions $C(R^\omega)$ both carry algebraic as well as coalgebraic structure. This suggests the presence of a more abstract description in terms of bialgebras for a distributive law [2,26]. However, it is not clear what the involved monad is, and as discussed after Def. 14 it is also not clear how open linear systems can be viewed as coalgebras over a category of algebras. We do not exclude the possibility of a bialgebraic modeling of open linear systems, but we have to leave it as future work.

References

1. M. Atiyah and I.G. MacDonald. *Introduction to Commutative Algebra.* Addison-Wesley series in mathematics, Westview Press 1994.
2. Bartels, F. On Generalised Coinduction and Probabilistic Specification Formats. Ph.D. thesis, Vrije Universiteit Amsterdam, 2004.
3. M.M. Bonsangue, J.J.M.M. Rutten, and A. Silva. Coalgebraic logic and synthesis of Mealy machines. In *Proceedings of FoSSaCS 2008*, volume 4962 of *Lecture Notes in Computer Science*, pages 231–245, Springer 2008.
4. M.M. Bonsangue, S. Milius, and A. Silva. Sound and complete axiomatizations of coalgebraic language equivalence. *ACM Transactions on Computational Logic* 14(1): 7-57, 2013.
5. W.-K. Chen. On flow graph solutions of linear algebraic equations. *SIAM Journal on Applied Mathematics* 15(1): 136—142, 1967.
6. R. Crochiere and A. Oppenheim. Analysis of linear digital networks. In *Proceedings of IEEE* 4: 581—595, 1975.
7. A.L. Davis and R.M. Keller. Dataflow program graphs. *Computer* 15(2): 26–41, 1982.
8. D. Eisenbud. *Commutative Algebra: With a View Toward Algebraic Geometry. Graduate Texts in Mathematics Series*, Springer, 1995.
9. H.H. Hansen. Subsequential transducers: a coalgebraic perspective. *Information and Computation* 208(12): 1368–1397, 2010.
10. H.H. Hansen and J.J.M.M. Rutten. Symbolic synthesis of Mealy machines from arithmetic bitstream functions. *Scientific Annals of Computer Science* 20: 97–130, 2010.
11. G. Kahn. The semantics of a simple language for parallel programming. *Information Processing* 74: 471–475, 1974.
12. R.M. Keller and P. Panangaden. Semantics of networks containing indeterminate operators. In *Seminar on Concurrency*, volume 197 of *Lecture Notes in Computer Science*, pages 479–496, Springer 1984.
13. R.M. Keller and P. Panangaden. Semantics of digital networks containing indeterminate modules. *Distributed Computing* 1(4): 235–245, 1986.
14. T.Y. Lam. Lectures on Modules and Rings. Graduate Texts in Math., Springer, 1999.
15. J. Lambek. A fixpoint theorem for complete categories. *Mathematische Zeitschrift* 103(2):151–161, 1968.
16. S.J. Mason. Feedback theory: Some properties of linear flow graphs. In *Proceedings of IRE 41*, pages 1144–1156, 1953.

17. J.L. Massay and M.K. Sain. Codes, Automata, and Continuous Systems: Explicit Interconnections. In *IEEE Trans. Automatic Control 12(6)*, pages 644–650, 1967.
18. J.L. Massay and M.K. Sain. Inverses of Linear Sequential Circuits. In *IEEE Trans. Comput. 17*, pages 330–337, 1968.
19. S. Milius. A sound and complete calculus for finite stream circuits. In *Proceedings of LICS 2010*, pages 421–430, IEEE, 2010.
20. K.K. Parhi and Y. Chen. Signal flow graphs and data flow graphs. In *Handbook of Signal Processing Systems*, pages 791–816, Springer, 2010.
21. J.J.M.M. Rutten. Universal coalgebra: a theory of systems. *Theoretical Computer Science* 249(1): 3–80, 2000.
22. J.J.M.M. Rutten. A coinductive calculus of streams. *Mathematical Structures in Computer Science* 15(1): 93–147, 2005.
23. J.J.M.M. Rutten. A tutorial on coinductive stream calculus and signal flow graphs. *Theoretical Computer Science* 343(3): 443–481, 2005.
24. J.J.M.M. Rutten. Rational streams coalgebraically. *Logical Methods in Computer Science* 3(9): 1–22, 2008.
25. M.B. Scherba and R.P. Roesser. Computation of the transition matrix of a linear sequential circuit. In *IEEE Trans. Computers 22(4)*, pages 427–428, 1973.
26. D. Turi and G. Plotkin. Towards a mathematical operational semantics. In *Proceedings of LICS 1997*, pages 280–291. IEEE Computer Society, 1997.
27. M. Vidyasagar. Control System Synthesis. A Factorization Approach. The *MIT Press*, 1988.
28. H.S. Wilf. generatingfunctionology. *Academic Press, Inc.*, Second edition, 1992.