Internal Report 2012-2013-07 July 2013

Universiteit Leiden
Opleiding Informatica

Linear Stream Circuits to Rational Streams

There and Back Again

Tobias Kappé

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Leiden University

Linear Stream Circuits and Rational Streams

There and Back Again

Author:
Tobias Kappé

Supervisors:
Dr. M. Bonsangue
J. Rot

July 1, 2013

Contents

1 Introduction

Ordered series of data are not new to computer science. A particular occurence is the output behavior of a transition system after each (discrete) transition. One might be interested in proving that this behavior is equivalent to some other system, possibly structured in a different way. These infinite series encoding output behavior, studied from the perspective of coalgebra, are called *streams* and the system we use for reasoning about them is called *stream calculus* [1, 6].

Of special interest are transformations of streams: how do we define them, and what properties do they preserve? Is it perhaps possible to convey these transformations graphically? In this work, we consider two existing approaches to define transformations on streams: *behavioral differential equations* and *stream circuits*. Moreover, we discuss a number of cases where a back-and-forth translation between the two methods is possible, extending a known correspondence from [2].

Section 2 is used to give an brief overview of the notation and conventions we will be using.

In Section 3, we review systems of streams called *stream automata*. We also consider a proof technique called *coinduction* that is useful for proving the equivalence of two streams. One can describe transformations on streams using *behavioral differential equations*. The existence of unique solutions to these equations for some classes of streams is proven. We then define a number of operators using behavioral differential equations and consider the structure they impose on the set of streams. At the end, we distinguish a subset of streams called *rational streams*.

Moving on to Section 4, we use a special kind of hypergraph called a *stream circuit* to sketch transformations on streams. A class of stream circuits called *linear stream circuits* appear to encode functions that multiply by a rational stream when given a fixed interpretation. We will derive a method that explicitly derives this rational stream for a given linear stream circuit. Furthermore, we note that an appropriate linear stream circuit can be derived from a rational stream and review a method for this derivation. There is also a first pass at deriving an equation from an alternative interpretation of a linear stream circuit. In the final subsection, we make a brief observation about simulating a linear stream circuit efficiently.

2 Preliminaries

A *binary relation* on a set X is a set $R \subseteq X \times X$. We write xRy if and only if $(x, y) \in R$. A *string* s over a set A is a finite sequence of elements in A, $s = a_0a_1...a_n$. By A^* we mean the set of all strings consisting of elements in A. If s and t are strings over A, then st is their *concatenation*, also a member of A^*. If s is a string over A and $p, q \in A$ we write $s[p := q]$ for the string s in which every occurence of p is replaced by q.

When treating functions as objects in context where domain and codomain are obvious, we may abbreviate our notation by using *lambda notation*, i.e. $s : \mathbb{N} \to \mathbb{N}, a \mapsto a + 1$ becomes $\lambda a.(a + 1)$. Let f and g be functions. By $f \circ g$ we denote the composition $\lambda x.f(g(x))$. We write $f \times g$ for their product $\lambda\langle x, y\rangle.\langle f(x), g(y)\rangle$ and $\langle f, g\rangle$ for their pairing $\lambda x.\langle f(x), g(x)\rangle$ (assuming they have the same domain). We say that f and g *commute* when $f \circ g = g \circ f$. A diagram as in Figure 1 is said to be *commutative* when $r \circ p = s \circ q$. A function with the empty set as domain is called a *nullary* function, functions of signature $S^n \to S$ are *operators* on S.

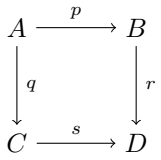


Figure 1: A commutative diagram

A *signature* is a tuple $\langle \Sigma, a\rangle$ such that Σ is a set and a is a function $\Sigma \to \mathbb{N}$ called the *arity function*. For $f \in \Sigma$, we call $a(f)$ the *arity* of f. If X is a set, then by $\Sigma_n(X)$ we denote the *language of expressions* of depth n over Σ. Inductively, for $n = 0$ we have $\Sigma_0(X) = X$ and for $n > 0$:

$$\Sigma_n(X) = \{f(x_1, ..., x_i) : f \in \Sigma,\ x_1, ..., x_i \in \Sigma_{n-1}(X),\ i = a(f)\} \cup \Sigma_{n-1}(X)$$

$\Sigma_*(X)$ is then the language of expressions of abritrary depth, that is the union of all $\Sigma_n(X)$.

A *group* is a tuple $\langle G, +, 0\rangle$ where $0 \in G$ and $+$ is an associative binary operator on G for which $0 + x = x + 0 = x$. Additionally, every element $x \in G$ has an *additive inverse* $-x \in G$ such that $x + -x = 0$. If $+$ is commutative, i.e. $x + y = y + x$ for all $x, y \in G$, then G is an *abelian* group.

A *ring* is a tuple $\langle R, +, \cdot, 0, 1\rangle$ such that $\langle R, +, 0\rangle$ is an abelian group. Furthermore, $\cdot$ is an operator on R for which $1 \cdot x = x = x \cdot 1$ and that distributes over $+$, i.e. $x \cdot (y + z) = x \cdot y + x \cdot z$ and $(x + y) \cdot z = x \cdot z + y \cdot z$. If $\cdot$ is commutative then R is a *commutative ring*. A *multiplicative inverse* of x is an element $x^{-1} \in R$ such that $x \cdot x^{-1} = 1$. If every element of a commutative ring R except 0 has a multiplicative inverse, then R is a *field*. A *ring homomorphism* between two rings $\langle R, +_R, \cdot_R\rangle$ and $\langle S, +_S, \cdot_S\rangle$ is a function $h : R \to S$ such that $h(x +_R y) = h(x) +_S h(y)$ and $h(x \cdot_R y) = h(x) \cdot_S h(y)$. A bijective ring homomorphism is an *isomorphism*.

3 Streams

In this section, we shall define and discuss *streams*. A stream is best viewed as an infinite series of which we can inspect the first element or advance the stream one position. Contrary to for example, mathematical series, we will seldom refer to individual elements in a stream by their index. Instead we will define streams using *behavioral differential equations*, in which we specify the initial value of a stream and the behaviour of its remainder with relation to itself. This approach requires us to employ a proof paradigm called *coinduction* (co-induction) when we want to prove equality of two streams.

Much of the terminology and notation has been borrowed from [1]. Our notation differs in that our streams do not necessarily consist of elements in $\mathbb{R}$. We also rigorously prove the existence of unique solutions to behavioral differential equations before using them to define operators. We recommend [1] for an in-depth discussion of streams and their applications in other branches of mathematics. For an interesting application of stream calculus and coinduction to Moessner's theorem, we refer the reader to [5].

3.1 Stream automata

If we consider a stream as an object consisting of an initial value and a remaining stream, the notion of a *stream automaton* arises naturally. We use the term *automaton* here because discarding the initial value of a stream in order to obtain the tail can be seen as a transition from one stream to another.

Indeed, a stream automaton is one example of a more general notion called an F-coalgebra [3] where F is a functor $FX = U \times X$ for some fixed set U. F-coalgebra's are known to resemble transition systems and are therefore often used when analysing them in an abstract fashion. A full treatment of F-coalgebra's requires familiarity with category theory and is beyond the scope of this work. Instead, we refer the reader to [3] for an introduction.

Definition 1. A *stream automaton* is a quadruple $\langle S, U, O, T\rangle$ where S is the *set of states* and U is the *output set*, O is a function $S \to U$ called the *output function* and T is a function $S \to S$ called the *transition function*.

As is common with these constructs, we will often refer to a stream automaton $\langle S, U, O, T\rangle$ by simply writing S. We will use the Greek letters $\sigma, \tau, \rho, ...$ to refer to the elements of S. If $\sigma \in S$ is such an element, we will write $\sigma(0)$ for $O(\sigma)$ and σ' for $T(\sigma)$ when no ambiguity arises with respect to which automaton σ belongs to. In analogy to classical calculus, we will refer to σ' as the *derivative* of σ. Note that we can build a stream automaton for any set U if we construct S from U as the set of series in U.

Definition 2. Let U be a set. We define the set of U-streams, U^ω, to be as follows:

$$U^\omega = \{f : f \text{ is a function } \mathbb{N} \to U\}$$

We can thus build a stream automaton $\langle U^\omega, U, O, T\rangle$, where we define

$$O : U^\omega \to U, \sigma \mapsto \sigma(0) \quad T : U^\omega \to U^\omega, \sigma \mapsto \lambda n.\sigma(n+1)$$

In the sequel, whenever we refer to the stream automaton U^ω, it will have the output- and transition functions as defined above. We will see in Section 3.3 that stream automata built in this fashion have a property called *finality*, meaning that we can derive a unique structure-preserving map from any automaton with the same output set.

When we want to speak of a stream $\sigma \in U^\omega$, we could simply list a few of its elements: $\sigma = (\sigma(0), \sigma(1), \sigma(2), ...)$ and expect the reader to infer the further structure of the stream. For example, if $U = \{0, 1\}$, we might want to consider $\sigma = (1, 1, ...)$. A more precise and less ad-hoc method of defining such streams is to use a *behavioral differential equation*. These are not unlike differential equations in classical calculus: we specify one or more initial values, and then define the further progression in terms of these initial values only. In stream calculus, this comes down to giving a finite number of initial elements and an equation for the tail of the stream beyond those. Coming back to our example for σ, we could define it as $\sigma(0) = 1$, $\sigma' = \sigma$. Intuitively, the finite prefix $(1, 1, ...)$ conforms to this definition. We shall return to behavioral differential equations more formally in Section 3.3.

3.2 Coinduction

Suppose we want to demonstrate equality between two streams $\sigma, \tau \in U^\omega$ that are defined using behavioral differential equations. Recall that σ and τ are functions $\mathbb{N} \to U$, thus we need to show that for all $n \in \mathbb{N}$, $\sigma(n) = \tau(n)$. Unfortunately, we usually do not have a direct formula for either stream, so we cannot expand the definition $\sigma(n)$ to an expression and show it to be equal to the definition of $\tau(n)$. Neither do we have an expression for $\sigma(n+1)$ in terms of $\sigma(n)$, meaning that mathematical induction is not (directly) fit for this task either. Instead, we need the notion of a *bisimulation*.

Definition 3. Let S be a stream automaton. A *bisimulation* on S is a relation $R \subseteq S \times S$ such that whenever $\sigma\ R\ \tau$ we know that $\sigma(0) = \tau(0)$ and $\sigma'\ R\ \tau'$.

If for some streams σ, τ there exists a stream bisimulation R such that $\sigma\ R\ \tau$, we will write $\sigma \sim \tau$. The stream bisimulation is, in essence, what allows us to use mathematical induction to prove equality between streams: if we know that two streams are bisimilar, their first elements must coincide; their derivatives are also bisimilar, thus *their* first elements (the second of the original streams) are also equal, et cetera.

Theorem 1 (Coinduction)**.** *Let U^ω be a stream automaton and $\sigma, \tau \in U^\omega$ such that $\sigma \sim \tau$. Then $\sigma = \tau$.*

Proof. Denote with $\sigma^{(n)}$ the n-th derivative of σ and let R be the bisimulation such that $\sigma \; R \; \tau$. We know that $\sigma^{(0)} \; R \; \tau^{(0)}$. Suppose that $\sigma^{(n)} \; R \; \tau^{(n)}$. By definition of a bisimulation, it follows that

$$\sigma^{(n+1)} = \left(\sigma^{(n)}\right)' \; R \; \left(\tau^{(n)}\right)' = \tau^{(n+1)}$$

By induction, we now know that for all $n \in \mathbb{N}$, $\sigma^{(n)} \; R \; \tau^{(n)}$ and since R is a bismulation it follows that for all $n \in \mathbb{N}$, $\sigma(n) = \tau(n)$. Thus we conclude $\sigma = \tau$. □

We have purposely restricted our coinduction theorem to stream automata of the form U^ω. If we were to consider it for general stream automata S, a counterexample is easily constructed. Suppose we take $S = \{\sigma, \tau\}$ with $\sigma(0) = \tau(0) = 0$ and $\sigma' = \sigma$ as well as $\tau' = \tau$; we can immediately see that $\sigma \sim \tau$ by the bisimulation $\{\langle \sigma, \tau \rangle\}$, even though σ and τ are not equal. In fact, we will show in the next section that the finality of U^ω is what makes the relation $\sim$ coincide with the diagonal.

For now, when we are in a situation where equality between two streams $\sigma, \tau \in U^\omega$ is to be proven, we simply need to find an appropriate relation $R \subseteq U^\omega \times U^\omega$ with $\sigma \; R \; \tau$ and show that R is a bisimulation. Equality then follows from this theorem. As a degenerate case, we present the following lemma:

Lemma 1. *Let $\sigma, \tau \in U^\omega$ be two streams for which $\sigma(0) = \tau(0)$ and $\sigma' = \tau'$. Then $\sigma = \tau$.*

Proof. Consider the bisimulation $R = \mathrm{id}_S \cup \{\langle \sigma, \tau \rangle\}$. □

This lemma is a first glimpse that we may specify our bisimulations in a more concise fashion. The set $\{\langle \sigma, \tau \rangle\}$ was not a bisimulation by itself, but in taking the union with the diagonal we have created a bisimulation. We shall encounter more instances of such smaller relations called *bisimulation-up-to* [4] in Section 3.5.

3.3 Homomorphisms

Another useful notion is that of a *homomorphism* between two stream automata, which we can use to demonstrate the existence of unique solutions to behavioral differential equations. Informally speaking, a homomorphism between two stream automata having the same output set is a mapping between their elements that preserves the transition and output functions.

Definition 4. Let $\langle S_1, U, O_1, T_1 \rangle$ and $\langle S_2, U, O_2, T_2 \rangle$ be two stream automata. A *homomorphism* from S_2 to S_1 is a function $h : S_2 \to S_1$ such that for all $\sigma \in S_1$, $O_2(\sigma) = O_1(h(\sigma))$ and $h(T_2(\sigma)) = T_1(h(\sigma))$. This means the diagram in Figure 2 commutes. As usual, a bijective homomorphism h shall be referred to as an isomorphism.

$$\begin{array}{ccc} S_2 & \xrightarrow{\langle T_2, O_2 \rangle} & S_2 \times U \\ \downarrow{\scriptstyle h} & & \downarrow{\scriptstyle h \times \mathrm{id}} \\ S_1 & \xrightarrow{\langle T_1, O_1 \rangle} & S_1 \times U \end{array}$$

Figure 2: Commutative diagram for Definition 4

It is easily verfied that the composition of two homomorphisms is again a homomorphism. If we have a stream automaton S_1 with output set U such that for all stream automata S_2 with the same output set there exists a unique homomorphism $S_2 \to S_1$, we will refer to S as a *U-final automaton*. Final automata will turn out to be useful later on, as the uniqueness of the homomorphisms into them will allow us to solve behavioral differential equations uniquely. We can also build a final automaton for any given output set U.

Theorem 2. *For all sets U, $\langle U^\omega, U, O, T \rangle$ is a U-final automaton.*

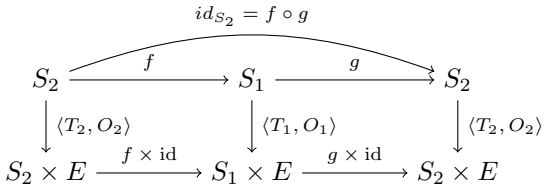


Figure 3: Commutative diagram for Theorem 3

Proof. Consider an automaton $\langle S, U, O_1, T_1 \rangle$. We define $h : S \to U^\omega$ as follows:

$$h(\sigma) = (O(\sigma), O(T(\sigma), O(T(T(\sigma))), ...)$$

The function h is now a homomorphism by construction. What remains to show is that h is unique. Suppose that $g : S \to U^\omega$ is another homomorphism. We take the following relation

$$R = \{\langle h(\sigma), g(\sigma) \rangle : \sigma \in \mathcal{S}\}$$

To show that this relation is a bisimulation, consider an element $\langle h(\sigma), g(\sigma) \rangle \in R$. We easily verify that $O(g(\sigma)) = O_1(\sigma) = O(h(\sigma))$ by virtue of h and g being homomorphisms. As for stream derivatives, we know that

$$T(g(\sigma)) = g(T_1(\sigma)) \; R \; h(T_1(\sigma)) = T(h(\sigma))$$

The fact that for all $\sigma \in S$, $h(\sigma) = g(\sigma)$ now follows by coinduction. □

In a sense, the automaton U^ω is the *only* U-final automaton that can be distinguished:

Theorem 3. *For any output set U, there is exactly one U-final automaton up to isomorphism.*

Proof. As we have seen in Theorem 2, we can construct at least one final automaton for any set U. What remains to show is that two final automata are isomorphic. Consider such automata $\langle S_1, U, O_1, T_1 \rangle$ and $\langle S_2, U, O_2, T_2 \rangle$. The definition of finality now provides two unique homomorphisms $f : S_1 \to S_2$ and $g : S_2 \to S_1$. Composition yields another homomorphism $f \circ g : S_2 \to S_2$. But the identity function id_{S_2} is trivially a homomorphism — by uniqueness it then follows that $f \circ g = \mathrm{id}_{S_2}$ (c.f. Figure 3). Similarly, we prove that $g \circ f = \mathrm{id}_{S_1}$. The function f is a bijective homomorphism: an isomorphism. □

We are now ready to state a more general and stronger form of Theorem 1.

Corollary 1. *Let $\langle S, U, O, T \rangle$ be a U-final automaton. Then $\sigma \sim \tau$ implies $\sigma = \tau$.*

Proof. Let $R \subseteq S \times S$ be the bisimulation for which $\sigma \; R \; \tau$ and let h be the isomorphism $S \to U^\omega$. We construct the relation $R_h = \{\langle h(\rho_1), h(\rho_2) \rangle : (\rho_1, \rho_2) \in R\}$. We can check that R_h is a bisimulation by considering a pair $\langle h(\rho_1), h(\rho_2) \rangle \in R$.

$$h(\rho_1)(0) = O(\rho_1) = O(\rho_2) = h(\rho_2)(0)$$

$$h(\rho_1)' = h(T(\rho_1)) \; R \; h(T(\rho_2)) = h(\rho_2)'$$

By Theorem 1 we derive that $h(\sigma) = h(\tau)$ and because h is a bijection, $\sigma = \tau$. □

One can easily construct a stream automaton with a finite number of streams in which bisimulation implies equality, showing that the converse of the above is not true in general.

3.4 Behavioral differential equations

We can use the U-final automaton U^ω to show that certain behavioral differential equations over terms in U^ω have unique solutions. This result is particularly convenient, because it allows us to define operators of arbitrary arity using behavioral differential equations. Our method differs from the one used in [1, 6] in the way we define differential equations. It is abstract enough to allow for all behavioral differential equations used in the sequel to have unique solutions. To get a feeling for this method, we first derive a special case and then proceed with the more general theorem.

Lemma 2. *Let U^ω be a stream automaton with $u \in U$. There is a unique stream $\sigma \in U^\omega$ satisfying $\sigma(0) = u$ and $\sigma' = \sigma$.*

Proof. Consider the stream automaton $\langle \{\tilde{\sigma}\}, U, O, T\rangle$. We define $O(\tilde{\sigma}) = u$ and $T(\tilde{\sigma}) = \tilde{\sigma}$. Because U^ω is a U-final automaton, there exists a unique homomorphism $h : \{\tilde{\sigma}\} \to U^\omega$. We calculate that $h(\tilde{\sigma})(0) = O(\tilde{\sigma}) = e$ and $h(\tilde{\sigma})' = h(T(\tilde{\sigma})) = h(\tilde{\sigma})$. Thus $h(\tilde{\sigma})$ satisfies the behavioral differential equation for σ. Consider a stream $\tau \in U^\omega$ that satisfies the behavioral differential equation as well. Then τ and σ are equal by Lemma 1. □

The above proof gives us an idea of the general technique for proving the existence of unique solutions to behavioral differential equations. We create a stream automaton in which the transitions of the streams behave syntactically equivalent to our equations. The actual solution to the equation is then obtained through the unique homomorphism to U^ω. In the theorem below, we generalize this method to allow for systems of mutually dependent behavioral differential equations of arbitrary arity. First, we need a more precise definition of what constitutes a system of behavioral differential equations.

Definition 5. Let $\langle U^\omega, U, O, T\rangle$ be a stream automaton and (Σ, a) a signature. A system of *behavioral differential equations* is a function $\lambda : \Sigma_1(U \times V \times V) \to U \times \Sigma_*(V)$ where

1. V is a (possibly infinite) set of variables, disjoint from Σ.
2. If $x \in \Sigma_1(U \times V_1 \times V_2)$ with $V_1, V_2 \subseteq V$, then $\lambda(x) \in U \times \Sigma_*(V_1 \cup V_2)$.
3. If $\lambda(x) = \langle \tilde{e}, y\rangle$, with $\tilde{e}, e \in E$ and $v_1, v_2, u_1, u_2 \in V$, then
$$\lambda(x[\langle e, v_1, v_2\rangle := \langle e, u_1, u_2\rangle]) = \langle \tilde{e}, y[v_1 := u_1, v_2 := u_2]\rangle$$

A *solution* to such a system is a pair $\langle \mathcal{F}, [\![-]\!]\rangle$ where $\mathcal{F}$ is a set of functions in which every symbol $f \in \Sigma$ with $n = a(f)$ has a function $\hat{f} \in \mathcal{F}$ of type $(U^\omega)^n \to U^\omega$ and $[\![-]\!]$ is a function $\Sigma_1(U^\omega) \to U^\omega$ such that $[\![f(\sigma_1, \sigma_2, ..., \sigma_n)]\!] = \hat{f}(\sigma_1, \sigma_2, ..., \sigma_n)$ for all $\sigma_1, \sigma_2, ..., \sigma_n \in U^\omega$ and $f \in \Sigma$.

This definition restricts the format of differential equations, but not in ways that unnecessarily restrict their flexibility. The first restriction simply guarantees that we will always have enough variables, and that their substitution will not inadvertently overwrite function symbols. The second restriction makes sure that the right-hand side of a transition does not introduce any variables that were not present in the input. The last restriction requires that all variables are treated in the same manner, making them interchangable.

We can think of the first component of $\lambda(x)$ as the initial value of an equation, while the second component specifies the transition. In reference to Lemma 2, we could give a (partial) specification of λ by $\lambda(\sigma) = \langle e, \sigma\rangle$, with $\sigma \in \Sigma$ a nullary function symbol – we will get back to this when we reprove Lemma 2 making use of such a λ.

Theorem 4. *Let $\langle U^\omega, U, O, T\rangle$ be the final stream automaton and $\lambda : \Sigma_1(U \times V \times V) \to U \times \Sigma_*(V)$ be a system of behavioral differential equations. Then there exists a unique solution $\langle \mathcal{F}, [\![-]\!]\rangle$.*

Proof. We build the stream automaton $\langle \Sigma_*(U^\omega), U, o, t\rangle$ as follows. If $s \in \Sigma_0(U^\omega) = U^\omega$, then $o(s) = O(s)$ and $t(s) = T(s)$. If $s = f(s_1, s_2, ..., s_n)$ is an expression of depth $m > 0$, then $s_1, s_2, ..., s_n$ are expressions of depth lesser than m. Inductively, we then know $o(s_i)$ and $t(s_i)$ for $i = 1, 2, ..., n$. We associate the variable v_i with s_i and v_i' with $t(s_i)$ for $i = 1, 2, ..., n$ — all variables being distinct. Let
$$\lambda(f(\langle o(s_1), v_1', v_1\rangle, ..., \langle o(s_n), v_n', v_n\rangle)) = \langle e, y\rangle$$
Then we define $o(s)$ to be e and $t(s)$ to be $y[v_1' := t(s_1), ..., v_n' := t(s_n), v_1 := s_1, ..., v_n := s_n]$. Because $y \in \Sigma_*(\{v_1', ..., v_n', v_1, ...v_n\})$ by definition of behavioral differential equations, and we replace all elements of V by elements of $\Sigma_*(U^\omega)$, $t(s)$ is now a value in $\Sigma_*(U^\omega)$. The choice of the variables is irrelevant, due to the third restriction on λ in Definition 5.

We have now properly defined a stream automaton on $\Sigma_*(U^\omega)$. By finality of U^ω, there exists a unique homomorphism $[\![-]\!] : \Sigma_*(U^\omega) \to U^\omega$. From Σ, we build $\mathcal{F}$: for each $f \in \Sigma$ with $a(f) = n$, we define the function $\hat{f} : (U^\omega)^n \to U^\omega$ by $\hat{f}(\sigma_1, \sigma_2, ..., \sigma_n) = [\![f(\sigma_1, \sigma_2, ..., \sigma_n)]\!]$.

It can be quickly seen that $\langle \mathcal{F}, [\![-]\!]\rangle$ is a unique solution to λ by construction, provided that we restrict the domain of $[\![-]\!]$ to $\Sigma_1(U^\omega)$. □

As an example of a use of Theorem 4, we can apply it to prove Lemma 2. Let (Σ, a) be a signature with $\Sigma = \{\tilde{\sigma}\}$ and $a(\tilde{\sigma}) = 0$. Since $\Sigma_1(E \times V \times V) = (E \times V \times V) \cup \{\tilde{\sigma}\}$ we can define $\lambda(\langle e, v', v\rangle) = \langle e, v'\rangle$ for all $e \in E$ and $v', v \in V$, as well as $\lambda(\tilde{\sigma}) = \langle e, \tilde{\sigma}\rangle$. Theorem 4 then provides us with a solution $\langle\{\sigma\}, [\![-]\!]\rangle$ such that $[\![\tilde{\sigma}]\!] = \sigma$. If we recall that $[\![-]\!]$ is a homomorphism $\Sigma_*(U^\omega) \to U^\omega$, we can derive that $O(\sigma) = O([\![\tilde{\sigma}]\!]) = o(\tilde{\sigma}) = e$ and $T(\sigma) = T([\![\tilde{\sigma}]\!]) = [\![t(\tilde{\sigma})]\!] = [\![\tilde{\sigma}]\!] = \sigma$.

From this point on, all behavioral differential equations shall be given in the form used in Lemma 2. The operators derived from them should be considered those obtained by application of Theorem 4.

3.5 Operators

We are now ready to define a number of operators on streams, using behavioral differential equations. All of these are recalled from [1]. It should be noted that some are analogous to familiar operators on power series.

Most operators on streams inherently depend on the structure of their underlying set. In this section, when we speak of a stream automaton U^ω, we assume that the set U has a ring structure. Before proceeding, it is useful to explicitly state the inclusion of any ring U in its stream automaton U^ω.

Definition 6. Let U^ω be a stream automaton with $u \in U$. By $[u]$ we denote the stream satisfying $[u](0) = u$ and $[u]' = [0]$, with 0 being the zero element of U.

Whenever it is clear that u is an element of U^ω instead of U, we may write u for $[u]$. The simplest and most obvious operator on streams is the *pairwise sum*.

Definition 7. Let σ, τ be streams in the automaton U^ω. By $\sigma + \tau$ we denote the stream satisfying $(\sigma + \tau)(0) = \sigma(0) + \tau(0)$ and $(\sigma + \tau)' = \sigma' + \tau'$.

Note that in the equation specifying the initial value of $\sigma + \tau$, the first plus-symbol is the operator we are defininining and the second plus-symbol is the one received from the ring. We will use operator symbols like this interchangably whenever the context allows no confusion. This choice is further justified by the fact that when we overload an operator on U^ω in this way, it will usually inherit many properties from its counterpart on U.

We could have also defined the sum operator using a direct formula like $(\sigma + \tau)(n) = \sigma(n) + \tau(n)$; in practice we will see that having a coinductive definition allows for simpler proofs. Furthermore, it is perfectly possible to define operators on streams that do not have a (known) direct formula, as we will see later on in this section.

Lemma 3. *$\langle U^\omega, +\rangle$ forms an abelian group.*

Proof. Each of the components below is a bisimulation, rendering their union a bisimulation as well.

$$\begin{aligned}
&\{\langle \sigma + \tau, \tau + \sigma\rangle : \sigma, \tau \in U^\omega\} \\
&\cup \{\langle \sigma + (\tau + \rho), (\sigma + \tau) + \rho\rangle : \sigma, \tau, \rho \in U^\omega\} \\
&\cup \{\langle 0 + \sigma, \sigma\rangle : \sigma \in U^\omega\} \\
&\cup \{\langle \sigma + -\sigma, 0\rangle : \sigma \in U^\omega\}
\end{aligned}$$

Where by $-\sigma$ we denote the stream satisfying $(-\sigma)(0) = -\sigma(0)$ and $(-\sigma)' = -\sigma'$. □

These properties are so familiar that we will often omit references to the above lemma when making use of them.

3.5.1 The convolution product

We could lift the multiplication operator from the ring to a multiplication operator on streams in a pointwise manner, like we have done with addition. It is however much more interesting to go with an initially more counterintuitive notion of multiplication called the *convolution product*. Surprisingly, we will see that many known properties of multiplication are shared with this new operator.

Definition 8. Let $\sigma, \tau \in U^\omega$ with U^ω a stream automaton. By $\sigma \times \tau$ we denote the stream satisfying $(\sigma \times \tau)(0) = \sigma(0) \times \tau(0)$ and $(\sigma \times \tau)' = \sigma' \times \tau + \sigma(0) \times \tau'$.[1] We shall write $\sigma^0 \equiv 1$ and $\sigma^{n+1} \equiv \sigma \times \sigma^n$.

In Definition 8, we implicitly assume that convolution product takes precedence over the pairwise sum. The following familiar properties hold for all $\sigma \in U^\omega$: (their proofs are obvious)

$$\sigma \times 1 = \sigma = 1 \times \sigma \quad \text{and} \quad \sigma \times 0 = 0 = 0 \times \sigma \tag{1}$$

In order to reason with the convolution product in our bisimulations, the notion of a *bisimulation-up-to* is instrumental. A bisimulation-up-to effectively allows us to specify a smaller relation that satisfies certain requirements, and build a bisimulation from there. The presence of a bisimulation-up-to then again implies equality.

Definition 9. Let $\langle U^\omega, U, O, T\rangle$ be a stream automaton and $R \subseteq U^\omega \times U^\omega$ be a relation. We say that R is a *bisimulation-up-to* if it contains the diagonal, and for all $\sigma, \tau \in U^\omega$ with $\sigma \; R \; \tau$, it holds that $\sigma(0) = \tau(0)$ and $\sigma' = \sigma_1 + ... + \sigma_n$ as well as $\tau' = \tau_1 + ... + \tau_n$ such that $\sigma_i \; R \; \tau_i$ for $i = 1, ..., n$. If σ' and τ' are related as such, we shall write $\sigma' \; \Sigma R \; \tau'$.

Theorem 5. *Let $\langle U^\omega, U, O, T\rangle$ be a stream automaton and R a bisimulation-up-to. If $\sigma \; R \; \tau$, then $\sigma = \tau$.*

Proof. Let $\bar{R} \subseteq S \times S$ be the smallest relation containing R and the identity, and if $\sigma_0 \; \bar{R} \; \tau_0$ as well as $\sigma_1 \; \bar{R} \; \tau_1$, then $\sigma_0 + \sigma_1 \; \bar{R} \; \tau_0 + \tau_1$. We claim that $\bar{R}$ is a bisimulation. To see this, consider $\sigma \; \bar{R} \; \tau$. If this is due to $\sigma \; R \; \tau$, then $\sigma(0) = \tau(0)$ and $\sigma' = \sigma_1 + ... + \sigma_n$ as well as $\tau' = \tau_1 + ... + \tau_n$ with $\sigma_i \; R \; \tau_i$ for $i = 1, ..., n$. By induction on the number of terms, it follows that $\sigma' \; \bar{R} \; \tau'$. If $\sigma \; \bar{R} \; \tau$ due to $\sigma = \tau$, then we are done immediately. Lastly, if $\sigma \; \bar{R} \; \tau$ because $\sigma = \sigma_0 + \sigma_1$ and $\tau = \tau_0 + \tau_1$ for $\sigma_0 \; \bar{R} \; \tau_0$ and $\sigma_1 \; \bar{R} \; \tau_1$, then $\sigma' = \sigma_0' + \sigma_1' \; \bar{R} \; \tau_0' + \tau_1' = \tau'$. Since $\sigma \; \bar{R} \; \tau$ and $\bar{R}$ is a bisimulation, $\sigma = \tau$ follows by coinduction. □

It is very conceivable that this notion of bisimulation-up-to be generalized to other operators, like the convolution product, or other manners of closure. In fact, such a generalisation has been carried out in [4]. For our purposes, however, the pairwise sum will suffice. With bisimulation-up-to in our toolbox, we can prove some other familiar identities.

Lemma 4. *Pairwise sum distributes over convolution product: for all $\sigma, \tau, \rho \in U^\omega$, $\sigma \times (\rho + \tau) = \sigma \times \tau + \sigma \times \rho$ and $(\sigma + \tau) \times \rho = \sigma \times \rho + \tau \times \rho$.*

Proof. Consider the relation $R = \{\langle \sigma \times (\rho + \tau), \sigma \times \rho + \sigma \times \tau\rangle : \sigma, \tau, \rho \in U^\omega\}$. Initial values are equal, and we move on to their derivatives:

$$\begin{aligned}(\sigma \times (\rho + \tau))' &= \sigma' \times (\rho + \tau) + \sigma(0) \times (\rho' + \tau') \\ &\; \Sigma R \; \sigma' \times \rho + \sigma' \times \tau + \sigma(0) \times \rho' + \sigma(0) \times \tau' \\ &= \sigma' \times \rho + \sigma(0) \times \rho' + \sigma' \times \tau + \sigma(0) \times \tau' \\ &= (\sigma \times \rho + \sigma \times \tau)'\end{aligned}$$

The case for right-distribution is proven analogously. □

Lemma 5. *Convolution product is associative, that is, for all $\sigma, \tau, \rho \in U^\omega$, $\sigma \times (\tau \times \rho) = (\sigma \times \tau) \times \rho$.*

Proof. Consider the relation $R = \{\langle \sigma \times (\tau \times \rho), (\sigma \times \tau) \times \rho\rangle : \sigma, \tau, \rho \in U^\omega\}$. Initial values are immediately equal and we calculate the derivatives:

$$\begin{aligned}(\sigma \times (\tau \times \rho))' &= \sigma' \times (\tau \times \rho) + \sigma(0) \times (\tau' \times \rho + \tau(0) \times \rho') \\ &= \sigma' \times (\tau \times \rho) + \sigma(0) \times (\tau' \times \rho) + \sigma(0) \times (\tau(0) \times \rho') \\ &\; \Sigma R \; (\sigma' \times \tau) \times \rho + (\sigma(0) \times \tau') \times \rho + (\sigma(0) \times \tau(0)) \times \rho' \\ &= (\sigma' \times \tau + \sigma(0) \times \tau') \times \rho + (\sigma(0) \times \tau(0)) \times \rho' \\ &= (\sigma \times \tau)' \times \rho + (\sigma \times \tau)(0) \times \rho' \\ &= ((\sigma \times \tau) \times \rho)'\end{aligned}$$

This allows us to write $\sigma \times \tau \times \rho$ without parentheses. □

[1] It is worth noting that the above definition of convolution product still conforms to Definition 5 in spite of the presence of $\sigma(0)$. This is because we implicitly introduced two function symbols g and 0 of input-arity one and zero respectively. For all $\langle e, v_0, v_1\rangle \in X$, we define $\lambda(g(\langle e, v_0, v_1\rangle)) = \langle e, 0\rangle$ and $\lambda(0) = \langle 0, 0\rangle$. In our solution for this system of behavioral differential equations, we will implicitly end up with $[\cdot] \circ O$ as the solution to g, which we then ignore.

Our nomenclature for the convolution product suggests that it has an inverse, i.e. for a stream σ we can derive a stream σ^{-1} such that $\sigma \times \sigma^{-1} = 1$. This is true for streams containing values from a field and which have a non-zero initial value – streams beginning with zero can never be multiplied to obtain a stream not starting with zero. If we start with $\sigma \times \sigma^{-1} = 1$, we can easily derive that $\sigma^{-1}(0)$ must be $\sigma(0)^{-1}$. As for derivatives, we know that $(\sigma \times \sigma^{-1})' = 0$, thus $\sigma(0) \times (\sigma^{-1})' + \sigma' \times \sigma^{-1} = 0$, meaning that $(\sigma^{-1})' = \sigma(0)^{-1} \times -\sigma' \times \sigma^{-1}$. We can thus define our convolution inverse:

Definition 10. Let $\sigma \in U^\omega$. By σ^{-1} we denote the stream satisfying $\sigma^{-1}(0) = \sigma(0)^{-1}$ and $(\sigma^{-1})' = \sigma(0)^{-1} \times -\sigma' \times \sigma^{-1}$ when $\sigma(0) \neq 0$. If $\sigma(0) = 0$, we define $\sigma^{-1} = 0$. As common, we will write $\sigma^{-1} \equiv \frac{1}{\sigma}$.

This construction is not unlike finding the multiplicative inverse in a polynomial ring. The approach of deriving the definition for an operator based on its desired behavior is worth stressing. When working with streams, one will often find this method to yield desirable results, as it does now:

Lemma 6. *Let $\sigma \in U^\omega$ with $\sigma(0) \neq 0$. Then $\sigma \times \sigma^{-1} = 1$.*

Proof. Initial values are equal by construction, thus we check the derivatives:

$$\begin{aligned}(\sigma \times \sigma^{-1})' &= \sigma' \times \sigma^{-1} + \sigma(0) \times (\sigma^{-1})' \\ &= \sigma' \times \sigma^{-1} + \sigma(0)^{-1} \times \sigma(0) \times -\sigma' \times \sigma^{-1} \\ &= \sigma' \times \sigma^{-1} + -\sigma' \times \sigma^{-1} = 0\end{aligned}$$

Equality now follows by Lemma 1. □

3.5.2 The shuffle product

Another operator that is worth considering is that of the *shuffle product*. The reader may have observed that the definition of the pairwise sum is very similar to the sum rule in classical calculus. Whereas the definition of the convolution product does not look at all like the product rule, the shuffle product does.

Definition 11. Let $\sigma, \tau \in U^\omega$. By $\sigma \otimes \tau$ we denote the stream satisfying $(\sigma \otimes \tau)(0) = \sigma(0) \times \tau(0)$ and $(\sigma \otimes \tau)' = \sigma' \otimes \tau + \sigma \otimes \tau'$.

Though the shuffle product is distinct from the convolution product, its behavior is very similar. The following properties are obviously true:

$$\sigma \otimes 1 = \sigma = 1 \otimes \sigma \quad \text{and} \quad \sigma \otimes 0 = 0 = 0 \otimes \sigma \tag{2}$$

Once more, their proofs follow from the obvious bisimulation-up-to. Other familiar identities are:

Lemma 7. *The shuffle product distributes over the pairwise sum: for all $\sigma, \tau, \rho \in U^\omega$, $\sigma \otimes (\rho + \tau) = \sigma \otimes \rho + \sigma \otimes \tau$ and $(\sigma + \tau) \otimes \rho = \sigma \otimes \rho + \tau \otimes \rho$.*

Proof. Consider the relation $R = \{\langle \sigma \otimes (\rho + \tau), \sigma \otimes \rho + \sigma \otimes \tau \rangle : \sigma, \tau, \rho \in U^\omega\}$. Initial values are equal;

$$\begin{aligned}(\sigma \otimes (\rho + \tau))' &= \sigma \otimes (\rho' + \tau') + \sigma' \otimes (\rho + \tau) \\ &\quad \Sigma R\ \sigma \otimes \rho' + \sigma \otimes \tau' + \sigma' \otimes \rho + \sigma' \otimes \tau \\ &= \sigma \otimes \rho' + \sigma' \otimes \rho + \sigma \otimes \tau' + \sigma' \otimes \tau \\ &= (\sigma \otimes \rho + \sigma \otimes \tau)'\end{aligned}$$

□

Lemma 8. *The shuffle product is associative: for all $\sigma, \tau, \rho \in U^\omega$, $\sigma \otimes (\tau \otimes \rho) = (\sigma \otimes \tau) \otimes \rho$.*

Proof. Consider the relation $R = \{\langle \sigma \otimes (\tau \otimes \rho), (\sigma \otimes \tau) \otimes \rho \rangle : \sigma, \tau, \rho \in U^\omega\}$. Initial values are equal;

$$\begin{aligned}(\sigma \otimes (\tau \otimes \rho))' &= \sigma' \otimes (\tau \otimes \rho) + \sigma \otimes (\tau \otimes \rho' + \tau' \otimes \rho) \\ &= \sigma' \otimes (\tau \otimes \rho) + \sigma \otimes (\tau \otimes \rho') + \sigma \otimes (\tau' \otimes \rho) && \text{[Lemma 7]} \\ &\quad \Sigma R\ (\sigma' \otimes \tau) \otimes \rho + (\sigma \otimes \tau') \otimes \rho + (\sigma \otimes \tau) \otimes \rho' \\ &= (\sigma' \otimes \tau + \sigma \otimes \tau') \otimes \rho + (\sigma \otimes \tau) \otimes \rho' && \text{[Lemma 7]} \\ &= ((\sigma \otimes \tau) \otimes \rho)'\end{aligned}$$

We will therefore write $\sigma \otimes \tau \otimes \rho$ without parentheses. □

3.6 Stream automata as rings

In this section, we will review the operators defined in the previous section. We shall consider two ring structures on U^ω, one given by the convolution product and the other by the shuffle product. The two ring structures we consider are in fact isomorphic for some U. For a full proof of their isomorphism, we refer the reader to Appendix A.

One central idea that we need is that we can transform a stream $\sigma = (s_0, s_1, ...)$ into a stream $\tau = (0, s_0, s_1, ...)$ when we multiply σ by a constant stream. In fact, this stream returns often enough that it warrants its its own symbol.

Definition 12. $X \in U^\omega$ is the stream satisfying $X(0) = 0$ and $X' = [1]$.

If taking a stream derivative is like calculating the derivative of a function in classical calculus, then multiplying by X and reinstating the first value would be our analogue of integration.

Theorem 6 (Fundamental Theorem)**.** *For all $\sigma \in U^\omega$, $\sigma = \sigma(0) + X \times \sigma'$*

Proof. Initial values are trivially equal. As for derivatives, consider

$$\begin{aligned}(\sigma(0) + X \times \sigma')' &= \sigma(0)' + X' \times \sigma' + X(0) \times \sigma'' && \text{[Definition 7 and 8]}\\ &= 0 + 1 \times \sigma' + 0 \times \sigma'' && \text{[Definition 6 and 12]}\\ &= \sigma'\end{aligned}$$

Equality now follows by Lemma 1. □

In fact, we can regard the fundamental theorem as a manifestation of Lambek's lemma [3], stating that the operation of a final coalgebra (or initial algebra) is necessarily a bijection.

Lemma 9. *If U is a commutative ring, then the convolution product on U^ω is commutative, that is, for all $\sigma, \tau \in U^\omega$, $\sigma \times \tau = \tau \times \sigma$.*

Proof. Consider the relation $R = \{\langle \sigma \times \tau, \tau \times \sigma \rangle : \sigma, \tau \in U^\omega\}$. We claim that R is a bisimulation-up-to. If $\sigma \times \tau \; R \; \tau \times \sigma$, then their initial values are obviously equal. We now consider their derivatives:

$$\begin{aligned}(\sigma \times \tau)' &= \sigma' \times \tau + \sigma(0) \times \tau'\\ &= \sigma' \times (\tau(0) + X \times \tau') + \sigma(0) \times \tau' && \text{[Fundamental theorem]}\\ &= \sigma' \times \tau(0) + \sigma' \times X \times \tau' + \sigma(0) \times \tau'\\ &\; \Sigma R \; \tau(0) \times \sigma' + \tau' \times X \times \sigma' + \tau' \times \sigma(0)\\ &= \tau(0) \times \sigma' + \tau' \times (X \times \sigma' + \sigma(0))\\ &= \tau(0) \times \sigma' + \tau' \times \sigma && \text{[Fundamental theorem]}\\ &= (\tau \times \sigma)'\end{aligned}$$

□

This observation, in conjunction with those from the previous section, allows us to state the following:

Theorem 7. *If $\langle U, +, \times \rangle$ is a (commutative) ring, then so is $\langle U^\omega, +, \times \rangle$.*

Proof. We know that $\langle U^\omega, + \rangle$ is a abelian group by Lemma 3. Associativity of the convolution product was shown in Lemma 5 and distributivity of the pairwise sum over convolution product in Lemma 4. The neutral element of the convolution product is found in Equation 1. This proves that U^ω is again a ring. If U is a commutative ring, then the convolution product is commutative by Lemma 9, making U^ω a commutative ring once more. □

Another ring structure on U^ω is given by the pairwise sum combined with the shuffle product, as we will motivate now.

Lemma 10. *If U is a commutative ring then the shuffle product is commutative in U^ω: for all $\sigma, \tau \in U^\omega$, $\sigma \otimes \tau = \tau \otimes \sigma$.*

Proof. Proven using the bisimulation-up-to $\{\langle \sigma \otimes \tau, \tau \otimes \sigma \rangle : \sigma, \tau \in U^\omega\}$. □

Theorem 8. *If $\langle U, +, \times\rangle$ is a (commutative) ring, then so is $\langle U^\omega, +, \otimes\rangle$*

Proof. As mentioned in Theorem 7 above, we know that $\langle U^\omega, +\rangle$ is an abelian group. By Lemma 7 we know that $\otimes$ distributes over $+$ and associativity of $\otimes$ was shown in Lemma 8. The shuffle product also has a neutral element, as seen in Equation 2. We can therefore conclude that $\langle U^\omega, +, \otimes\rangle$ is a proper ring. If U is a commutative ring, then so is U^ω with the shuffle product, as shown in Lemma 10. □

3.7 Classes of streams

Before proceeding, it is useful to distinguish a number of different kinds of streams. In this section, we will assume that we are working with streams from a commutative ring U^ω. This distinction is analogous to the one made on real numbers and is inspired by [5].

One obvious class of streams consists of *polynomial* streams.

Definition 13. A *polynomial* stream is a stream π such that $\pi^{(n)} = 0$ for certain n. If π is a polynomial stream then we may write the *degree* of π as $\deg(\pi) = n - 1$, which we leave undefined for the stream 0.

Intuitively, we see that this stream can only contain a finite number of nonzero elements, meaning that $\rho = (r_0, ..., r_{n-1}, 0, ...)$. We can therefore always repeatedly apply the fundamental theorem to write a polynomial stream as follows: $\rho = r_0 + r_1 \times X + ... + r_{n-1} \times X^{n-1}$. It is immediately clear that polynomial streams are closed under convolution product and pairwise sum, meaning that they form a proper subring of $\langle U, +, \times\rangle$.

Definition 14. A *rational* stream is a stream ρ such that $\rho = \pi \times \tau^{-1}$ for polynomial streams π and τ.

Rational streams are a proper superset of polynomial streams. Obviously, all polynomial streams are rational. Not all rational streams are polynomial; consider for example $\sigma = (1 - X)^{-1}$, which satisfies $\sigma(0) = 1$ and $\sigma' = ((1 - X)^{-1})' = (1 - X)^{-1} = \sigma$, meaning that $\sigma = (1, 1, 1, ...)$. Rational streams are once again closed under pairwise sum and convolution product, making them yet another subring of $\langle U, +, \times\rangle$.

4 Building functions on streams

The coinductive framework for streams makes it very natural to speak of them as though they really are natural *streams*: one element comes out, followed by an indefinite number of elements, ad infinitum. Just like streams of water, we could try to arrange them in network-like structures with nodes at which streams might join and/or split.

In this section we will explore *stream circuits* as a method to build functions on streams. Stream circuits are hypergraphs that obey a certain structure with respect to their hyperedges. If we associate each hyperedge type with a concrete function that takes one or more streams and yields one or more streams as well, we can interpret a given stream circuit as a function on U^ω, with U being a ring.

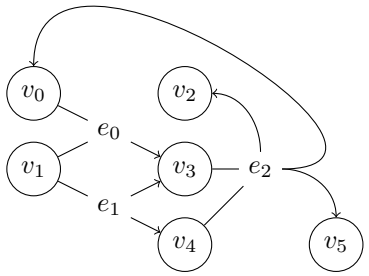
The notion of stream circuits arose from the world of signal processing, where they are known as *signal-flow graphs*. The connection to stream calculus was first made in [2]. We reiterate several results from this paper here more formally and further develop the correspondence between stream circuits and functions on streams.

4.1 Hypergraphs

We start out by defining a *hypergraph*. A hypergraph can be seen as a directed graph in which an edge can have more than one origin or destination; edges are thus made between words over the language of vertices.

Definition 15. A *hypergraph* $\mathcal{H}$ is a tuple (V, E) such that V and E are finite with $E \subseteq V^* \times V^*$. For each $e = \langle x, y\rangle \in E$, we write $i(e) = |x|$ and $o(e) = |y|$ for the number of originating, respectively ending, vertices of e. We call the elements of V the *vertices* of the hypergraph, while E contains the *hyperedges*.

Figure 4: An example of a hypergraph



As an example of a hypergraph, consider Figure 4. We see that e_0 is an edge with multiple inputs, while e_1 has multiple outputs. Cycles are allowed as well, as shown by e_2.

We cannot lift the notion of paths from directed graphs, because for a general hypergraph, a sequence of vertices may no longer uniquely determine a path. If we consider two vertices, there may be more than one way to reach the second from the first, because there may be more than one hyperedge connecting them. For example, the sequence $v_1v_3v_5$ of vertices in Figure 4 does not specify how we got from v_1 to v_3; was it through e_0 or e_1? We thus need to include edges in the path as well.

Definition 16. Let (V, E) be a hypergraph over some signature. We define a *hyperpath* or simply *path* in (V, E) to be a word $v_0e_0v_1e_1...v_n \in (V \cdot E)^* \cdot V$. For all $0 \leq j < n$, we know that $e_j = \langle i, o\rangle \in E$ and require $i = wv_jx$ and $o = yv_{j+1}z$ for some $w, x, y, z \in V^*$. A path $v_0e_0v_1e_1...v_n$ in which $v_0 = v_n$ is called a *cycle*.

It is also useful to have a notion of the in- and out-degree of vertices in a hypergraph. This is not to be confused with the input- and output-arity of an edge. The former refers to the amount of edges leaving (respectively entering) a vertex, while the latter refers to the number of vertices connected where an edge originates (respectively ends).

Definition 17. Let (V, E) be a hypergraph over some signature and $k \in V$. By $\text{in}(k)$ we denote the number of $\langle \kappa, i, o\rangle \in E$ such that $o = xky$, with $x, y \in V^*$. Similarly, we write $\text{out}(k)$ for the number of $\langle \kappa, i, o\rangle \in E$ such that $i = xky$ with $x, y \in V^*$.

Turning to Figure 4, we see that $\text{in}(v_3) = \text{out}(v_1) = 2$. When a vertex of a hypergraph has an in-degree of zero, it will be called an *input vertex*; likewise, vertices with an out-degree of zero will be called *output-vertices*.

4.2 Stream circuits

It is intuitively clear that a hypergraph can be used to model a network of streams, splitting and joining at hyperedges. For our intended purpose, we now need to properly restrict the hypergraph structure and relate the edges to stream transformations. We need vertices to unambiguously receive their information from a single source; there also needs to be a well-defined output vertex.

Definition 18. A *stream circuit* $\mathcal{S}$ over a signature $\langle \Sigma, a\rangle$ is a tuple $\langle V, E, l\rangle$ such that

1. $\langle V, E\rangle$ is a hypergraph with exactly one output-vertex and for all $v \in V$, $\text{in}(v) \leq 1$.
2. Σ contains a symbol c with $a(c) = 1$.
3. l is a function $E \to \Sigma$ such that for all $\langle x, y\rangle \in E$: $a(l(x, y)) = |x|$.
4. For all $\langle x, y\rangle \in E$, if $l(x, y) = c$, then $|y| = 2$, otherwise $|y| = 1$.

Of course, when we are speaking of one stream circuit in particular, it is unnecessarily complex to write it out entirely every time. We now establish some conventions for our stream circuit diagrams. If we look at the stream circuit $\mathcal{S}_1$ in Figure 5, we can easily see the input- and output-vertices μ and ν. There are three hyperedges: one from $\mu\rho_2$ to ρ_0 of type $+$, one from ρ_0 to $\rho_1\nu$ of type c and another

from ρ_1 to ρ_2 of type R. Vertices are distinguished from edges in that lines leaving them start with an arrow shaft and lines ending in them have an arrowhead. We will also use Greek letters exclusively to label vertices. The signature of $\mathcal{S}_1$ implicitly contains R and $+$; we can observe that $a(+) = 2$ as well as $a(R) = 1$. We may also abbreviate our notation for c-edges by simply letting two edges leave from the originating node, as shown in Figure 6.

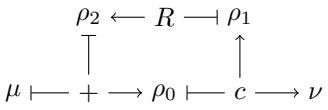


Figure 5: An example of a stream circuit

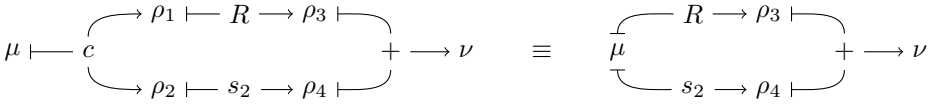


Figure 6: An abbreviated stream circuit, without c-edges

In order to treat a stream circuit as a function, we will need to interpret each function in the signature. This will yield a defining equation for each vertex in the stream circuit. The solution to the defining equation for the output node in terms of the input node can then be viewed as a function.

Definition 19. Let $\mathcal{S} = \langle V, E, l\rangle$ be a stream circuit over $\langle \Sigma, a\rangle$. An *interpretation* of $\mathcal{S}$ over a stream automaton U^ω associates for each symbol $f \in \Sigma$ except c an operator on U^ω by the same name and of matching arity. We translate the hyperedges into equations as follows:

- $e = \langle v, u_1 u_2\rangle$ (thus $l(e) = c$) maps to $u_1 = v$ and $u_2 = v$.
- $e = \langle v_1 ... v_n, u\rangle$ maps to $u = f(v_1, ..., v_n)$ where f is the function associated with $l(e)$.

An expression of ν in terms of $\mu_1, ..., \mu_n$ alone which is derived from these equations is a *solution* to $\mathcal{S}$.

It is immediately clear that any interpretation of a stream circuit that does not contain loops will give us a solution: we start with the equation for ν and substitute until we have only variables representing input vertices on the left hand side. In this case, the stream circuit is essentially an expression tree.

The presence of loops does, however, not preclude the possibility of a solution. We can interpret Figure 5 as follows: the symbol $+$ is associated with the pairwise sum and R with the function $\sigma \mapsto \sigma \times X$. The equations we get using Definition 19 are then:

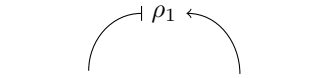
$$\begin{aligned} \rho_0 &= \mu + \rho_2 \\ \rho_2 &= \rho_1 \times X \\ \nu &= \rho_0 \\ \rho_1 &= \rho_0 \end{aligned}$$

Substituting ρ_0 we get $\nu = \mu + \rho_2$; if we substitute ρ_2 we are left with $\nu = \mu + \rho_1 \times X$, which is equivalent to $\nu = \mu + \nu \times X$. Inspecting initial value and derivative, we obtain the behavioral differential equation $\nu(0) = \mu(0)$ and $\nu' = \mu' + \nu$. Looking back at Definition 5 and interpreting ν as a unary function symbol, this behavioral differential equation gives a unique solution by Theorem 4. In fact, if we assume that we are working with streams from U^ω with U a field, we can derive this solution: $(1 - X) \times \nu = \mu$, therefore $\nu = \frac{\mu}{1-X}$.

Unfortunately, not all stream circuits with a feedback loop have a solution. If we consider the stream circuit in Figure 7 with the same interpretation as before, we get the following equations:

$$\begin{aligned} \rho_0 &= \mu + \rho_1 \\ \rho_1 &= \rho_0 \end{aligned}$$

Figure 7: A stream circuit without a solution



$$\nu = \rho_0$$

From this system, we get the equation $\nu = \mu + \mu + ...$, which makes no sense. Furthermore, we can derive $\mu = \rho_0 - \rho_1 = 0$, which is even less desirable. It turns out that the delay introduced by the R-hyperedge in Figure 5 is crucial to get a solution that makes sense.

4.3 Linear stream circuits to functions

One particularly interesting type of stream circuit is that of the *linear stream circuit*. In this section, we will introduce this class of stream circuits and show that we can uniquely derive a function $f : (U^\omega)^n \to U^\omega$ for each linear stream circuit, where n is the number of input vertices.

Definition 20. A *linear stream circuit* $\langle V, E, l\rangle$ is a stream circuit over the the signature $\langle \Sigma, a\rangle$ where

1. $\Sigma = \{c, +, R\} \cup \{s_r : r \in \mathbb{R}\}$ and $a(+) = 2$ while $a(f) = 1$ for all other symbols.
2. If $v_1e_1v_2...v_n$ is a cycle, then $l(e_i) = R$ for some $1 \leq i < n$.

The first requirement for a linear stream circuit essentially tells us about the signature we are working with; the second tells us that beyond a certain length, all paths need to pass through an R-hyperedge. Turning to Figure 5, we see that this stream circuit conforms to Definition 20, but the stream circuit that did not give a solution (Figure 7) violates the latter condition.

Definition 21. Let $\mathcal{S}$ be a linear stream circuit. By the *default interpretation* of $\mathcal{S}$ we mean the following interpretation. The $+$-gate is associated with the pairwise sum, the R-gate is associated with the function $\sigma \mapsto \sigma \times X$, and the s_r gates are associated with $\sigma \mapsto r \times \sigma$.

In the theorem below, we work out a solution method that can be applied to linear stream circuits when given the default interpretation. This is a more explicit construction of the theorem as found in [2]. In this instance, we require that the final stream automaton in the interpretation originates from a field. We will relax this requirement later on and see that a solution exists even without it.

Theorem 9. *Given a linear stream circuit $\mathcal{S}$ along with the default interpretation over U^ω, where U is a field, we can derive a solution.*

Proof. Let $\rho_1, ..., \rho_n$ be the *intermediate variables* associated with the inputs to the R-nodes of $\mathcal{S}$ and let $\nu_1, ..., \nu_m$ be the *input variables*, representing the input nodes of $\mathcal{S}$. Given the equations obtained from our interpretation, we can derive for each intermediate variable an equation in terms of intermediate and input variables: we simply substitute in the equation until all variables are from this set. This is possible, if we realise that substitution means travelling backwards along a path in the hypergraph, and all paths in a linear stream circuit eventually end at an input node or encounter an R-hyperedge, meaning we substitute an intermediate variable. We are now left with the following system of equations:

$$\begin{array}{lcccccccccccc}
\rho_1 & = & r_{1,1} \times \rho_1 \times X & + & \cdots & + & r_{n,1} \times \rho_n \times X & + & s_{1,1} \times \mu_1 & + & \cdots & + & s_{m,1} \times \mu_m \\
\rho_2 & = & r_{1,2} \times \rho_1 \times X & + & \cdots & + & r_{n,2} \times \rho_n \times X & + & s_{1,2} \times \mu_1 & + & \cdots & + & s_{m,2} \times \mu_m \\
\vdots & & \vdots & & & & \vdots & & \vdots & & & & \vdots \\
\rho_n & = & r_{1,n} \times \rho_1 \times X & + & \cdots & + & r_{n,n} \times \rho_n \times X & + & s_{1,n} \times \mu_1 & + & \cdots & + & s_{m,n} \times \mu_m
\end{array}$$

in which $r_{i,j}$ and $s_{i,j}$ are the products of scalars encountered along the path. We can denote this system

more compactly using matrix multiplication:

$$\begin{bmatrix} r_{1,1} \times X - 1 & \cdots & r_{n,1} \times X & s_{1,1} & \cdots & s_{m,1} \\ \vdots & & \vdots & \vdots & & \vdots \\ r_{1,n} \times X & \cdots & r_{n,n} \times X - 1 & s_{1,n} & \cdots & s_{m,n} \end{bmatrix} \times \begin{bmatrix} \rho_1 \\ \vdots \\ \rho_n \\ \mu_1 \\ \vdots \\ \mu_m \end{bmatrix} = 0 \tag{3}$$

We can see that in this matrix, we have $n+m$ columns and n rows. The rows are linearly independent because a linear combination of i rows among them inescapably ends up with i columns having a term that is not multiplied by X; no two rows can therefore be linearly combined to form a row already in this matrix. Moreover, no row is a scaling of another for a similar reason: each row has a unique column containing a term not multiplied by X.

It now is tempting to claim that we can solve this system by writing the n intermediate variables in terms of the m input variables by applying Gaussian elimination. This would, however, be incorrect as U^ω itself is not a field — we cannot divide by X, for example. We are thus required to explicitly show that for this matrix, Gaussian elimination needs not divide by such elements.

Suppose we start with the matrix in Equation 4. We can clear the first column from terms multiplied by X easily; we substract the first row multiplied by $\frac{r_{1,2}}{r_{1,1}}$ from the second row and so forth for the other rows. We can then divide the topmost row by $r_{1,1} \times X - 1$, yielding:

$$\begin{bmatrix} 1 & \cdots & \frac{r_{n,1} \times X}{r_{1,1} \times X - 1} & \frac{s_{1,1}}{r_{1,1} \times X - 1} & \cdots & \frac{s_{m,1}}{r_{1,1} \times X - 1} \\ \vdots & & \vdots & \vdots & & \vdots \\ q & \cdots & (r_{n,n} - q \cdot r_{n,1}) \times X - 1 & s_{1,n} - q \cdot s_{1,1} & \cdots & s_{m,n} - q \cdot s_{m,1} \end{bmatrix}$$

where $q = \frac{r_{1,n}}{r_{1,1}}$. The first column now containing scalars only, it is easy to eliminate them by subtracting an appropriate multiplication of the first row. If we carry out this process for the other rows, we end up with a matrix resembling:

$$\begin{bmatrix} 1 & \cdots & 0 & -\alpha_{1,1} & \cdots & -\alpha_{m,1} \\ \vdots & \ddots & \vdots & \vdots & & \vdots \\ 0 & \cdots & 1 & -\alpha_{1,n} & \cdots & -\alpha_{m,n} \end{bmatrix}$$

It is important to realise that the entries $\alpha_{i,j}$ now symbolize concrete streams rather than scalars, due to the operations performed on all rows. Looking back at Equation 4, we can now express each intermediate variable in terms of input variables only:

$$\rho_i = \alpha_{1,i} \times \mu_1 + \cdots + \alpha_{m,i} \times \mu_m \quad \text{for } i = 1, ..., m$$

If we then take the variable ν to represent the output node and derive an equation for it in terms of intermediate and input-nodes like we did for the intermediate nodes, we can substitute the equations for the intermediate nodes we just found. In doing so, we have obtained an equation for ν in terms of the input nodes $\mu_1, ..., \mu_m$ and have therefore found a solution for the default interpretation of $\mathcal{S}$. □

As hinted before, the requirement that U be a field is not necessary to prove the existence of a solution to our interpretation of a linear stream circuit.

Theorem 10. *Let $\mathcal{S}$ be a linear stream circuit with the default interpretation over U^ω, with U any ring. Then there exists a unique solution to this interpretation of $\mathcal{S}$.*

Proof. Consider the system of equations from Theorem 9. Taking the derivatives and initial values, we obtain the equations:

$$\left.\begin{array}{rcl} \rho_i(0) & = & s_{1,i} \times \mu_1(0) + \cdots + s_{m,i} \times \mu_m(0) \\ \rho_i' & = & r_{1,i} \times \rho_1 + \cdots + r_{n,i} \times \rho_n + s_{1,i} \times \mu_1' + \cdots + s_{m,i} \times \mu_m' \end{array}\right\} \text{for } i = 1, ..., n$$

These conform to Definition 5, meaning that by Theorem 4 there exist uniqe functions $f_i : (U^\omega)^m \to U^\omega$ such that for $i = 1, ..., n$ we have: $\rho_i = f_i(\mu_0, ..., \mu_m)$. As in Theorem 9, we can now express the output variable ν in terms of these functions, thus finding a unique solution. □

It follows that there is exactly one solution to the default interpretation of a linear stream circuit. For brevity, we may refer to it as the *default solution*.

Corollary 2. *The default solution of a linear stream circuit with m inputs can be expressed in the form $\nu = \tau_1 \times \mu_1 + ... + \tau_n \times \mu_m$, where τ_i is a rational stream (recall Definition 14) for $1 \leq i \leq m$.*

Proof. As can be seen in the proof of Theorem 9, each ρ_i can be expressed as $\rho_i = \alpha_{1,i} \times \mu_1 + ... + \alpha_{m,i} \times \mu_m$. In this equation, $\alpha_{1,i}, ..., \alpha_{m,i}$ are rational streams, because they are derived using the pairwise sum, convolution product and inverse of rational streams only. We also know that

$$\nu = p_1 \times \rho_1 \times X + ... + p_n \times \rho_n \times X + q_1 \times \mu_1 + ... + q_m \times \mu_m$$

for some scalars $p_1, ..., p_n$ and $q_1, ..., q_m$. Filling in $\rho_1, ..., \rho_n$ then gives:

$$\nu = \tau_1 \times \mu_1 + ... \tau_m \times \mu_m \quad \text{where} \quad \tau_j = p_j \times \alpha_{1,j} \times X + ... + p_n \times \alpha_{n,j} \times X + q_j \quad (1 \leq j \leq m)$$

in which $\tau_1, ..., \tau_m$ are rational streams because they are products and sums of rational streams only. □

The advantage of Theorem 9 with respect to Theorem 10 above is that it provides us with a solution constructively, allowing us to make statements about the output behavior of linear stream circuits when interpreted over U^ω where U is a field.

Corollary 3. *The default solution of a linear stream circuit preserves rationality of the input.*

Proof. Each linear stream circuit encodes a function $f(\mu_1, ..., \mu_m) = \tau_1 \times \mu_1 + ... + \tau_m \times \mu_m$ for rational streams $\tau_1, ..., \tau_m$. By closure of rational streams under pairwise sum and convolution product, it follows that $f(\mu_1, ..., \mu_m)$ must be rational if $\mu_1, ..., \mu_m$ are. □

4.4 Functions to stream circuits

In the preceding, we have derived functions on streams from stream circuits. In this section, we investigate whether we can go the other way around. Given a rational stream $\rho = \pi \times \tau^{-1}$ where π and τ are polynomial, we want to derive a linear stream circuit that represents the function $f(\mu) = \rho \times \mu$ under the interpretation as in Theorem 9. From here on, it is obvious how multi-variable functions could be translated to multiple-input linear stream circuits.

To do this, we first confine ourselves to multiplication by polynomial streams $\pi = r_0 + r_1 \times X + ... + r_n \times X^n$. Since multiplication by π distributes over the individual terms of the polynomial, it suffices to find a circuit that multiplies by each term before summing them up. Multiplication by scalars and X being built into linear stream circuits already, we quickly find the circuit depicted in Figure 8.

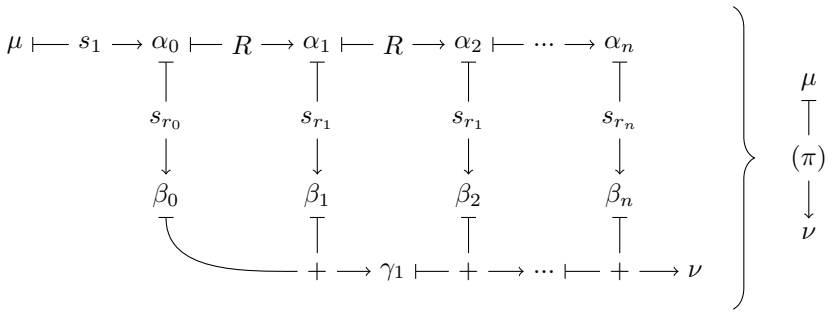


Figure 8: A linear stream circuit multiplying by the polynomial stream π and its abbreviation.

Tracing back the edges from each β-node, we see that $\beta_i = r_i \times X^i \times \mu$. Since ν is the sum of all such β_i, it follows that

$$\nu = r_0 \times \mu + r_1 \times X \times \mu + ... + r_n \times X^n \times \mu = \pi \times \mu$$

Suppose now that we want to *divide* the input stream by some constant polynomial stream τ. That would mean that we are looking for a stream circuit encoding equation $\nu = \tau^{-1} \times \mu$. Multiplying by τ

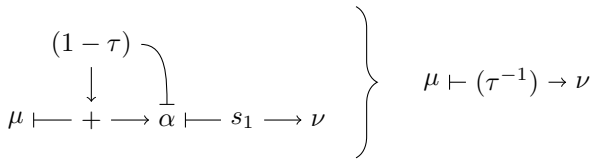


Figure 9: A circuit dividing by the polynomial stream τ, along with its abbreviation.

gives us $\tau \times \nu = \mu$. Defining $\hat{\tau} := \tau - 1$ and substituting for τ gives us $(\hat{\tau} + 1) \times \nu = \mu$ which can be rewritten to

$$\nu = \mu - \hat{\tau} \times \nu = \mu + (1 - \tau) \times \nu$$

This last equation is one we can encode in a linear stream circuit as in Figure 9. Obviously, these constructions can now be combined to create a linear circuit that computes the function $f(\mu) = \rho \times \mu$ for a *rational* stream $\rho = \pi \times \tau^{-1}$.

It should be noted that the number of R-gates used in this construction is exactly $\deg(\pi) + \deg(\tau)$. Another solution in [2] uses $\max(\deg(\pi), \deg(\tau))$ R-gates by collapsing the two circuits (one for multiplication, one for division) into one, thus avoiding the repeated calculation of $\mu \times X^i$-terms. We found this split solution to be a bit more instructive.

We are now in a position to state the central theorem of this section:

Theorem 11. *Every linear stream circuit implements a multiplication by a rational stream, and every function that multiplies by a rational stream can be encoded in a linear stream circuit.*

Proof. We have shown how to derive a function that multiplies by a rational stream in Theorem 9. In this section, we have shown that any multiplication by a rational stream can be encoded in a linear stream circuit. □

Corollary 4. *Every linear stream circuit $\mathcal{S}$ can be transformed into a linear stream circuit that provides the same default solution $\mathcal{S}'$. This new linear stream circuit first multiplies by a polynomial stream, then divides by it.*

Proof. We can derive the rational stream ρ from $\mathcal{S}$ as in Corollary 2 and then construct the linear stream circuit defined by ρ as instructed in this section. The resulting stream circuit $\mathcal{S}'$ will encode the same function as $\mathcal{S}$ and will have the right structure. □

4.5 Extending circuit solvability

There is another interpretation of a linear stream circuit that will yield unique solutions. If we interpret an R-edge by the function $\sigma \mapsto \sigma \otimes X$ we can still derive a solution. In order to do that, we will need the *shuffle inverse* operator.

Definition 22. Let $\sigma^{\underline{-1}}$ denote the *shuffle inverse* of σ, defined by $\sigma^{\underline{-1}}(0) = \sigma(0)^{-1}$ and $(\sigma^{\underline{-1}})' = \sigma^{\underline{-1}} \otimes -\sigma' \otimes \sigma^{\underline{-1}}$. When $\sigma(0) = 0$, we define $\sigma^{\underline{-1}} = 0$.

It is easily verified that $\sigma \otimes \sigma^{\underline{-1}} = 1$. Using this analogue of the convolution inverse operator, we can proceed to interpret and solve linear stream circuits like we have done in the previous section.

Lemma 11. *Given a linear stream circuit $\mathcal{S}$ along with the interpretation over U^ω, with U a field, of the pairwise sum for the symbol $+$, the function $\sigma \mapsto \sigma \otimes X$ for R and $\sigma \mapsto r \times \sigma$ for all s_r, we can derive a solution.*

Proof. We start out with the matrix-denoted system of equations:

$$\begin{bmatrix} r_{1,1} \otimes X - 1 & \cdots & r_{n,1} \otimes X & s_{1,1} & \cdots & s_{m,1} \\ \vdots & & \vdots & \vdots & & \vdots \\ r_{1,n} \otimes X & \cdots & r_{n,n} \otimes X - 1 & s_{1,n} & \cdots & s_{m,n} \end{bmatrix} \otimes \begin{bmatrix} \rho_1 \\ \vdots \\ \rho_n \\ \nu_1 \\ \vdots \\ \nu_m \end{bmatrix} = 0 \tag{4}$$

The solution is found using the same Gaussian elimination technique, whereby we never have to use a shuffle inverse of a stream starting with zero. □

That this interpretation also provides a solution is due to a property shared between the shuffle- and convolution product with X. We have seen before that $\sigma \times X = (0, \sigma(0), \sigma(1), ...)$, introducing a 'delay' in the stream. A quick calculation shows that $\sigma \otimes X = (0, 1 \cdot \sigma(0), 2 \cdot \sigma(1), ...)$. Our new interpretation of the R-gate thus introduced a suitable delay in the (distorted) values of σ.

4.6 Simulating a linear stream circuit

Given a linear stream circuit and an input stream σ, we might want to calculate its output stream (or rather, a finite prefix thereof) under the default solution. In this section, we present a short algorithm that can be used to achieve this.

Assume we start with some linear stream circuit with a single input, and an input stream σ of which we want to know the first n outputs. By Corollary 2, we can derive a rational stream $\rho = \pi \times \tau^{-1}$ such that the default solution to our circuit would be $f(\sigma) = \rho \times \sigma$. The initial value is trivially $\rho(0) \times \sigma(0)$. We set out to calculate $f(\sigma)(1)$. Let us first compute the derivative of ρ:

$$\begin{aligned}\rho' = (\pi \times \tau^{-1})' &= \pi(0) \times (\tau^{-1})' + \pi' \times \tau^{-1} \\ &= \pi(0) \times \tau(0)^{-1} \times -\tau' \times \tau^{-1} + \pi' \times \tau^{-1} \\ &= \frac{\pi' - \pi(0) \times \tau(0)^{-1} \times \tau'}{\tau}\end{aligned}$$

If we define $s(\pi, \tau) = \pi' - \pi(0) \times \tau(0)^{-1} \times \tau'$, then we can calculate the derivative of $\frac{\pi}{\tau} \times \sigma$:

$$\left(\frac{\pi}{\tau} \times \sigma\right)' = \left(\frac{\pi}{\tau}\right)' \times \sigma(0) + \frac{\pi}{\tau} \times \sigma' = \frac{s(\pi, \tau) \times \sigma(0)}{\tau} + \frac{\pi}{\tau} \times \sigma'$$

Using this, it is easy to calculate $f(\sigma)(1)$. From the structure of our expression for $\left(\frac{\pi}{\tau} \times \sigma\right)'$, it is apparent that taking yet another derivative will probably yield a very similar structure, one we will exploit in a moment. Consider the recurrent series defined by $\gamma_{n+1} = s(\gamma_n, \tau) + s(\pi, \tau) \times \sigma(n)$, $\gamma_0 = 0$. Our claim is:

$$\left(\frac{\pi}{\tau} \times \sigma\right)^{(n)} = \frac{\gamma_n}{\tau} + \frac{\pi}{\tau} \times \sigma^{(n)} \tag{5}$$

We prove Equality 5 by induction. For $n = 0$, it is trivial. For $n > 0$, we compute:

$$\begin{aligned}\left(\frac{\pi}{\tau} \times \sigma\right)^{(n+1)} &= \left(\left(\frac{\pi}{\tau} \times \sigma\right)^{(n)}\right)' \\ &= \left(\frac{\gamma_n}{\tau} + \frac{\pi}{\tau} \times \sigma^{(n)}\right)' && \text{[Induction hypothesis]} \\ &= \left(\frac{\gamma_n}{\tau}\right)' + \left(\frac{\pi}{\tau} \times \sigma^{(n)}\right)' \\ &= \frac{s(\gamma_n, \tau)}{\tau} + \left(\frac{\pi}{\tau} \times \sigma^{(n)}\right)' \\ &= \frac{s(\gamma_n, \tau)}{\tau} + \frac{s(\pi, \tau) \times \sigma^{(n)}(0)}{\tau} + \frac{\pi}{\tau} \times (\sigma^{(n)})' \\ &= \frac{s(\gamma_n, \tau) + s(\pi, \tau) \times \sigma(n)}{\tau} + \frac{\pi}{\tau} \times \sigma^{(n+1)} \\ &= \frac{\gamma_{n+1}}{\tau} + \frac{\pi}{\tau} \times \sigma^{(n+1)}\end{aligned}$$

This shows that we can calculate the first n outputs of the default solution to a linear stream circuit by calculating $\gamma_1, ..., \gamma_n$ and from those the initial values of Equality 5 for $i = 0, ..., n$.

As for complexity, take $m := \max(\deg(\pi), \deg(\tau))$. We know that $\deg(\gamma_i)$ will never exceed m as can easily be seen from its definition. To calculate γ_{i+1} from γ_i, one needs no more than $2m$ scalar multiplications. To calculate $f(\sigma)(i)$ a constant number of multiplications suffice. Calculating the first n values is therefore on the order $O(mn)$. The relatively small complexity of this algorithm appears due to the 'forgetfulness' of the gates in a linear stream circuit: only the R-gate needs to 'remember' previous input, exactly one element.

5 Discussion and Future Work

In this thesis, we discussed behavioral differential equations over streams in final stream automata. We explicitly proved the existence of unique solutions to these equations using an additional stream automaton of expressions. With this fact in hand, we showed that a ring structure on a set U carries over to U^ω and that commutativity is preserved. The structure of U^ω appears very similar to the ring of formal power series $U[\![X]\!]$, although the constructions of their operations follow different paradigms. Further investigation could show whether these two are isomorphic, and see whether results from formal power series have application within the context of streams.

We formalised the notation of stream circuits to make use of hypergraphs and distinguished the class of linear stream circuits, separating their syntactic structure from their semantic interpretation. We extended an existing [2] translation from linear stream circuits to rational streams to use linear algebra, a method explicit enough to be automated. We furthermore noted that this translation is unique and that a solution must exist even if the output set U is not a field. Future work might look into extensions of linear stream circuits to contain, for example, a multiplication gate or a gate that alternates incoming streams (zipper) as in [5].

Going the other way around, we showed how functions that multiply by a rational stream can be translated to linear stream circuits. Though our translation used more R-gates than an already known [2] method, we found this stepwise solution to be more instructive. In establishing the mutual correspondence between linear stream circuits and rational streams, we established a normal form for linear stream circuits.

Moreover, we discussed how our solution technique can be applied to linear stream circuit with an alternative interpretation of the R-gate, and briefly considered what makes our technique work on this alternative interpretation. These conditions could be formalised and used to generalize the approach to any conformant interpretation of the R-gate.

Lastly, we took an algorithmic approach at simulating the output of a linear stream circuit given an input stream, showing that the time-complexity of calculating a finite prefix of the output is linear in the size of this prefix and a parameter derived from the linear stream circuit. It is conceivable that the addition of a multiplication gate will significantly complicate an algorithm to simulate the output of the stream circuit.

Appendices

A Two ring structures on U^ω

In this appendix we look at stream automata U^ω where U is not just a ring, but also a vector space over $\mathbb{R}$. This extra assumption will allow us to construct an isomorphism [1] between the two ring structures on U^ω as reviewed in Section 3.6. In order to prove this isomorphism, we first need to consider an extension of the pairwise sum to an infinite number of operands.

A.1 The infinite sum operator

All operators as of yet had a finite number of operands. It is however perfectly possible for an operator to have an infinite number of arguments, as evidenced by the infinite sum. In order for this operator to remain well-defined, we must limit its domain.

Definition 23. Let $(\sigma_n)_{n=0}^\infty$ be a series of streams. If for all k, $\sum_{i=0}^\infty \sigma_i(k) < \infty$, then we call this series *summable*. For all summable series, we define the infinite sum $\sum_{i=0}^\infty \sigma_i$ to be the unique stream satisfying

$$\left(\sum_{i=0}^{\infty} \sigma_i\right)(0) = \sum_{i=0}^{\infty} \sigma_i(0) \quad \text{and} \quad \left(\sum_{i=0}^{\infty} \sigma_i\right)' = \sum_{i=0}^{\infty} \sigma_i'$$

Because we prefer our operator to be a total function, we define $\sum_{i=0}^\infty \sigma_i = 0$ when $(\sigma_n)_{n=0}^\infty$ is not summable. This distinction means we will have to take care to use the infinite sum operator on summable

series only [2].

The infinite sum behaves in the same manner as the pairwise sum does, with respect to operators we have previously encountered, as evidenced by the following lemma:

Lemma 12. *The following equalities hold for all summable families* $(\sigma_n)_{n=0}^{\infty}$ *and* $(\tau_n)_{n=0}^{\infty}$ *and streams* ρ*:*

$$\sum_{i=0}^{\infty} \sigma_i + \sum_{i=0}^{\infty} \tau_i = \sum_{i=0}^{\infty} (\sigma_i + \tau_i) \tag{6}$$

$$\rho \times \sum_{i=0}^{\infty} \sigma_i = \sum_{i=0}^{\infty} (\rho \times \sigma_i) \tag{7}$$

$$\rho \otimes \sum_{i=0}^{\infty} \sigma_i = \sum_{i=0}^{\infty} (\rho \otimes \sigma_i) \tag{8}$$

Proof. The proof of Equality 6 is very simple and therefore skipped. We give a proof for Equality 7. Consider the relation

$$R = \left\{ \left\langle \rho \times \sum_{i=0}^{\infty} \sigma_i,\ \sum_{i=0}^{\infty} \rho \times \sigma_i \right\rangle : \rho \in U^{\omega}, (\sigma_n)_{n=0}^{\infty} \text{ summable} \right\}$$

Initial values are equal, thus we calculate the derivatives:

$$\begin{aligned} \left(\rho \times \sum_{i=0}^{\infty} \sigma_i \right)' &= \rho' \times \sum_{i=0}^{\infty} \sigma_i + \rho(0) \times \sum_{i=0}^{\infty} \sigma_i' \\ \Sigma R \ &\sum_{i=0}^{\infty} \rho' \times \sigma_i + \sum_{i=0}^{\infty} \rho(0) \times \sigma_i' \\ &= \sum_{i=0}^{\infty} \rho' \times \sigma_i + \rho(0) \times \sigma_i' = \left(\sum_{i=0}^{\infty} \rho \times \sigma_i \right)' \end{aligned}$$

Equality now follows by Theorem 5. The proof for Equality 8 is very similar. □

The infinite sum allows us to extend Theorem 6 to include all elements of a stream.

Theorem 12. *For all streams* $\sigma \in U^{\omega}$*,* $\sigma = \sum_{n=0}^{\infty} \sigma(n) \times X^n$

Proof. We first need to prove that $(\sigma(n) \times X^n)_{n=0}^{\infty}$ is at all summable. This is indeed so; for any $k \geq 0$:

$$\left(\sum_{i=0}^{\infty} \sigma(i) \times X^i \right)(k) = \sigma(k) < \infty$$

Equality is proven by the obvious bisimulation. □

A.2 The isomorphism $\Lambda : U^{\omega} \to U^{\omega}$

In this section, we explicitly construct the isomorphism Λ between the two ring structures on U^{ω}, along with its inverse Γ. In order for this isomorphism to be properly defined, we will need to have a notion of multiplying a stream by a real number; we obtain this by assuming that U is a vector space over $\mathbb{R}$ and lifting the scalar product to U^{ω}. When $U = \mathbb{R}$, this notion happens to coincide with multiplication.

Theorem 13. *If* U *is a ring and a vector space over* $\mathbb{R}$*, then so is* U^{ω}*.*

[2] A critical reader may have already objected to our definition; strictly speaking, we cannot define an operator of infinite arity using the framework of Definition 5. We observe that the framework can be extended to allow for operators of (countably) infinite arity without too much trouble, and proceed without further proof.

Proof. We introduce the scalar product operator $\cdot : \mathbb{R} \times U^\omega \to U^\omega$ as follows: for all $x \in X$ and $\sigma \in U^\omega$, $(x \cdot \sigma)(0) = x \cdot \sigma(0)$ and $(x \cdot \sigma)' = x \cdot \sigma'$. This pairwise multiplication obviously satisfies the properties of the scalar product in a vector space. Further axioms of the vector space are satisfied from the fact that U^ω is a ring, by Theorem 7. □

In the remainder of this section, we will assume that U is a ring and a vector space for all U^ω we mention. One operator that returns often is that of the analytical derivative, as introduced in [1].

Definition 24. Let $\frac{d\cdot}{dX} : U^\omega \to U^\omega$ be the operator defined by $\frac{d\sigma}{dX} = (X \otimes \sigma')'$ for all $\sigma \in U^\omega$.

The notation for this operator is not coindidence. If $\sigma = (\sigma(0), \sigma(1), \sigma(2), ...)$ then $\frac{d\sigma}{dX} = (\sigma(1), 2 \cdot \sigma(2), 3 \cdot \sigma(3), ...)$, which is not unlike the derivative of a polynomial. It is this operator that will allow us to define an isomorphism between the two ring structures on U^ω.

Lemma 13. *The following equalities hold for all $\sigma \in U^\omega$, $e \in E$ and $n \in \mathbb{N}$:*

$$e \times \frac{d\sigma}{dX} = \frac{d(e \times \sigma)}{dX} \quad \textit{and} \quad \frac{dX^{n+1}}{dX} = (n+1) \cdot X^n \quad \textit{and} \quad \frac{d(X^n \times \sigma)}{dX} = \frac{d\sigma}{dX} \times X^n + \frac{dX^n}{dX} \times \sigma$$

Proof. The first equality is a direct consequence of the definition of $\frac{d\cdot}{dX}$. We prove the second equality by induction. For $n = 0$, it is easily calculated that the equality holds. For $n > 0$, observe:

$$\begin{aligned}
\frac{dX^{n+1}}{dX} &= (X \otimes (X^{n+1})')' \\
&= X^n + X \otimes (X^n)' \\
&= X^n + (X \times (X \otimes (X^n)')') && \text{[fundamental theorem]} \\
&= X^n + (X \times \frac{dX^n}{dX}) && \text{[definition of } \tfrac{d\cdot}{dX}\text{]} \\
&= X^n + (X \times n \cdot X^{n-1}) && \text{[induction hypothesis]} \\
&= (n+1) \cdot X^n
\end{aligned}$$

As for the last equality, we prove it by induction as well. For $n = 0$ and $n = 1$, it degenerates to the definition of $\frac{d\cdot}{dX}$. For $n > 1$, we derive:

$$\begin{aligned}
\frac{d(X^n \times \sigma)}{dX} &= (X \otimes (X^n \times \sigma)')' \\
&= \sigma \times X^{n-1} + (X \otimes (\sigma \times X^{n-1})')' && \text{[fundamental theorem]} \\
&= \sigma \times X^{n-1} + X \times \frac{d(\sigma \times X^{n-1})}{dX} && \text{[definition of } \tfrac{d\cdot}{dX}\text{]} \\
&= \sigma \times X^{n-1} + X \times \left(\frac{d\sigma}{dX} \times X^{n-1} + \frac{dX^{n-1}}{dX} \times \sigma \right) && \text{[induction hypothesis]} \\
&= \frac{d\sigma}{dX} \times X^n + \sigma \times X^{n-1} + \sigma \times X \times \frac{dX^{n-1}}{dX} \\
&= \frac{d\sigma}{dX} \times X^n + \sigma \times X^{n-1} + \sigma \times X \times (n-1) \cdot X^{n-2} && \text{[first equality]} \\
&= \frac{d\sigma}{dX} \times X^n + n \cdot X^{n-1} \times \sigma \\
&= \frac{d\sigma}{dX} \times X^n + \frac{dX^n}{dX} \times \sigma
\end{aligned}$$

□

Lemma 14. *For all summable series $(\sigma_n)_{n=0}^\infty$, it holds that*

$$\frac{d}{dX} \left(\sum_{n=0}^{\infty} \sigma_n \right) = \sum_{n=0}^{\infty} \frac{d\sigma_n}{dX}$$

Proof. We derive as follows:

$$\begin{aligned}
\frac{d}{dX}\left(\sum_{n=0}^{\infty}\sigma_n\right) &= \left(X \otimes \left(\sum_{n=0}^{\infty}\sigma_n\right)'\right)' \\
&= \sum_{n=0}^{\infty}\sigma_n' + X \otimes \sum_{n=0}^{\infty}\sigma_n'' \\
&= \sum_{n=0}^{\infty}(\sigma_n' + X \otimes \sigma_n'') && \text{[Lemma 12]} \\
&= \sum_{n=0}^{\infty}(X \otimes \sigma_n')' = \sum_{n=0}^{\infty}\frac{d\sigma_n}{dX} && \square
\end{aligned}$$

We can now prove the following, crucial lemma:

Lemma 15. *For all $\sigma, \tau \in U^\omega$, $\frac{d(\sigma\times\tau)}{dX} = \frac{d\sigma}{dX} \times \tau + \sigma \times \frac{d\tau}{dX}$.*

Proof.

$$\begin{aligned}
\frac{d(\sigma \times \tau)}{dX} &= \frac{d}{dX}\left(\sigma \times \sum_{i=0}\tau(i) \times X^i\right) && \text{[Theorem 12]} \\
&= \frac{d}{dX}\left(\sum_{i=0}\sigma \times \tau(i) \times X^i\right) && \text{[Lemma 12]} \\
&= \sum_{i=0}\frac{d}{dX}(\sigma \times \tau(i) \times X^i) && \text{[Lemma 14]} \\
&= \sum_{i=0}\left(\frac{d}{dX}(\sigma \times \tau(i)) \times X^i + \sigma \times \tau(i) \times \frac{dX^i}{dX}\right) && \text{[Lemma 13]} \\
&= \sum_{i=0}\left(\frac{d}{dX}(\sigma \times \tau(i)) \times X^i\right) + \sum_{i=0}\left(\sigma \times \tau(i) \times \frac{dX^i}{dX}\right) && \text{[Lemma 12]} \\
&= \sum_{i=0}\left(\frac{d\sigma}{dX} \times \tau(i) \times X^i\right) + \sum_{i=0}\left(\sigma \times \frac{d}{dX}(\tau(i) \times X^i)\right) && \text{[Lemma 13]} \\
&= \frac{d\sigma}{dX} \times \sum_{i=0}\left(\tau(i) \times X^i\right) + \sigma \times \sum_{i=0}\left(\frac{d}{dX}(\tau(i) \times X^i)\right) && \text{[Lemma 12]} \\
&= \frac{d\sigma}{dX} \times \tau + \sigma \times \frac{d}{dX}\left(\sum_{i=0}(\tau(i) \times X^i)\right) && \text{[Theorem 12, Lemma 14]} \\
&= \frac{d\sigma}{dX} \times \tau + \sigma \times \frac{d\tau}{dX} && \text{[Lemma 14]} \quad \square
\end{aligned}$$

We now have all machinery in place to fulfill our promise of deriving an isomorphism.

Theorem 14. *The rings $\langle U^\omega, +, \times\rangle$ and $\langle U^\omega, +, \otimes\rangle$ are isomorphic.*

Proof. Consider the function $\Lambda : U^\omega \to U^\omega$ defined as follows: $\Lambda(\sigma)(0) = \sigma(0)$ and $\Lambda(\sigma)' = \Lambda(\frac{d\sigma}{dX})$. From the bisimulation $\{\langle \Lambda(\sigma+\tau), \Lambda(\sigma) + \Lambda(\tau)\rangle : \sigma, \tau \in U^\omega\rangle\}$ it is easily proven that for all $\sigma, \tau \in U^\omega$, $\Lambda(\sigma + \tau) = \Lambda(\sigma) + \Lambda(\tau)$. We move on to prove that $\Lambda(\sigma \times \tau) = \Lambda(\sigma) \otimes \Lambda(\tau)$ using the relation $R = \{\langle \Lambda(\sigma \times \tau), \Lambda(\sigma) \otimes \Lambda(\tau)\rangle : \sigma, \tau \in U^\omega\}$. Initial values are obviously equal. As for the derivatives:

$$\begin{aligned}
(\Lambda(\sigma) \otimes \Lambda(\tau))' &= \Lambda(\sigma)' \otimes \Lambda(\tau) + \Lambda(\sigma) \otimes \Lambda(\tau)' \\
&= \Lambda\left(\frac{d\sigma}{dX}\right) \otimes \Lambda(\tau) + \Lambda(\sigma) \otimes \Lambda\left(\frac{d\tau}{dX}\right) \\
&\Sigma R\ \Lambda\left(\frac{d\sigma}{dX} \times \tau\right) + \Lambda\left(\sigma \times \frac{d\tau}{dX}\right)
\end{aligned}$$

$$= \Lambda\left(\frac{d\sigma}{dX} \times \tau + \sigma \times \frac{d\tau}{dX}\right)$$
$$= \Lambda\left(\frac{d(\sigma \times \tau)}{dX}\right) = \Lambda(\sigma \times \tau)' \qquad \text{[Lemma 15]}$$

We have now shown that Λ is at least a homomorphism from $\langle U^\omega, +, \times\rangle$ to $\langle U^\omega, +, \otimes\rangle$. What remains is to show that it is bijective, making it an isomorphism. Consider the function $\Gamma : U^\omega \to U^\omega$ defined by $\Gamma(\sigma)(0) = \sigma(0)$ and $\Gamma(\sigma)' = \Gamma(\sigma') - X \otimes \Gamma(\sigma)''$. Consider the relation $R = \{\langle \Lambda(\Gamma(\sigma)), \sigma\rangle : \sigma \in U^\omega\}$.

$$\begin{aligned}
\Lambda(\Gamma(\sigma))' &= \Lambda\left(\frac{d}{dX}(\Gamma(\sigma))\right) \\
&= \Lambda((X \otimes \Gamma(\sigma)')') \\
&= \Lambda(\Gamma(\sigma)' + X \otimes \Gamma(\sigma)'') \\
&= \Lambda(\Gamma(\sigma') - X \otimes \Gamma(\sigma)'' + X \otimes \Gamma(\sigma)'') \\
&= \Lambda(\Gamma(\sigma'))\ R\ \sigma'
\end{aligned}$$

Thus proving that $\Lambda(\Gamma(\sigma)) = \sigma$ for all $\sigma \in U^\omega$, showing that Λ is a bijection. □

References

[1] J. Rutten, *Elements of Stream Calculus (An Extensive Exercise in Coinduction)*. ENTCS vol. 45, pages 358–423, 2001.

[2] J. Rutten, *An Application of Stream Calculus to Signal Flow Graphs*. LNCS vol. 3188, pages 276-291, 2004.

[3] B. Jacobs and J. Rutten. *A tutorial on (co)algebras and (co)induction*. Bull. EACTS vol. 62, pages 222–259, 1997.

[4] J. Rot, M. Bonsangue and J.Rutten, *Coalgebraic bisimulation-up-to*. SOFSEM 2013: Theory and Practice of Computer Science, pages 369–381, Springer Berlin Heidelberg, 2013

[5] M. Niqui and J. Rutten, *Stream processing coalgebraically*. Science of Computer Programming, 2012, doi:10.1016/j.scico.2012.07.013.

[6] J. Rutten, *Behavioural differential equations: a coinductive calculus of streams, automata, and power series*. Theoretical Computer Science 308.1, pages 1–53, 2003.