On the final coalgebra of automatic sequences

Clemens Kupke[1] and Jan Rutten[2,3]

[1] University of Oxford
[2] Centrum Wiskunde & Informatica (CWI)
[3] Radboud University Nijmegen

Abstract. Streams are omnipresent in both mathematics and theoretical computer science. Automatic sequences form a particularly interesting class of streams that live in both worlds at the same time: they are defined in terms of finite automata, which are basic computational structures in computer science; and they appear in mathematics in many different ways, for instance in number theory. Examples of automatic sequences include the celebrated Thue-Morse sequence and the Rudin-Shapiro sequence. In this paper, we apply the coalgebraic perspective on streams to automatic sequences. We show that the set of automatic sequences carries a final coalgebra structure, consisting of the operations of head, even, and odd. This will allow us to show that automatic sequences are to (general) streams what rational languages are to (arbitrary) languages.

1 Introduction

The set of infinite sequences, or *streams*, over an alphabet A is defined by

$$A^\omega = \{\sigma \mid \sigma : \mathbb{N} \to A\}$$

In universal coalgebra [Rut00], which is a general theory of the behaviour of dynamical systems and infinite data types, the set of streams is the prototypical example of a *final coalgebra*, much in the same way as in universal algebra the set of natural numbers is the typical example of an initial algebra. The (final) coalgebra structure of A^ω is given by the isomorphism

$$A^\omega \to A \times A^\omega \qquad \sigma \mapsto (\sigma(0),\, \sigma')$$

which maps a stream σ to the pair consisting of its *head* or *initial value* $\sigma(0)$, and its *tail* or *stream derivative* σ', which is defined by

$$\sigma' = (\sigma(1), \sigma(2), \sigma(3), \ldots)$$

This elementary structure on A^ω gives rise to a surprisingly powerful collection of definition and proof methods. For instance, in so-called *stream calculus* [Rut05], one defines streams by means of *stream differential equations*, in close analogy to classical calculus in mathematics. Examples are

$$\sigma' = 3 \times \sigma \qquad \sigma(0) = 1$$

(with $3 \times \sigma = (3 \cdot \sigma(0), 3 \cdot \sigma(1), 3 \cdot \sigma(2), \ldots)$), which defines the stream $(3^0, 3^1, 3^2, 3^3, \ldots)$; and

$$\sigma' = \sigma \times \sigma \qquad \sigma(0) = 1$$

(where now $\times$ stands for convolution product), which defines the stream $(1, 1, 2, 5, 14, \ldots)$ of the so-called Catalan numbers. Furthermore, stream calculus comes along with a proof principle called *coinduction*, by which two streams can be shown to be equal by the construction of a suitable $\{\mathsf{head}, \mathsf{tail}\}$-*bisimulation relation*.

Streams are omnipresent in both mathematics and theoretical computer science. *Automatic sequences* [Fog02,AS03] form a particularly interesting class of streams that live in both worlds at the same time: they are defined in terms of finite automata, which are basic computational structures in computer science; and they appear in mathematics in many different ways, for instance in number theory. We will see a formal definition later but, in a nutshell, a stream is (2-)automatic if its n-th value is obtained by feeding the binary encoding of the number n into a Moore machine, and reading off the output value of the state thus reached. Examples of automatic sequences include the celebrated Thue-Morse sequence and the Rudin-Shapiro sequence.

In this paper, we set out to apply the coalgebraic perspective on streams to automatic sequences. As it happens, we shall not be using the afore-mentioned stream calculus, which is based on the standard final coalgebra structure of head and tail that we saw above. Instead, we will use a stream calculus that could be called *non-standard* because it is based on a different coalgebra structure on streams, which was recently introduced in [KR10]. This coalgebra structure is defined by

$$\langle \mathsf{head}, (\mathsf{even}, \mathsf{odd}) \rangle : A^\omega \to A \times (A^\omega)^2$$

where $\mathsf{head}(\sigma) = \sigma(0) \in A$ is again the initial value of σ and where

$$\mathsf{even}(\sigma) = (\sigma(0), \sigma(2), \sigma(4), \ldots) \qquad \mathsf{odd}(\sigma) = (\sigma(1), \sigma(3), \sigma(5), \ldots)$$

As we shall see, the coalgebra $(A^\omega, \langle \mathsf{head}, (\mathsf{even}, \mathsf{odd}) \rangle)$ is final among (a subclass of) the coalgebras of type $S \to A \times S^2$. Such coalgebras are known in the literature as 2-automata (with outputs in A); they are also called Moore machines.

The above final coalgebra structure on A^ω gives rise to a new type of stream calculus, in which streams can be defined by so-called $\{\mathsf{head}, \mathsf{even}, \mathsf{odd}\}$-stream differential equations. For instance,

$$\mathsf{head}(\sigma) = 0 \qquad \mathsf{even}(\sigma) = \mathsf{zeros} \qquad \mathsf{odd}(\sigma) = \mathsf{ones}$$

with $\mathsf{zeros} = (0, 0, 0, \ldots)$ and $\mathsf{ones} = (1, 1, 1, \ldots)$ defines the stream $\sigma = (0, 1, 0, 1, 0, 1, \ldots)$. Another, less trivial example is

$$\mathsf{head}(M) = 0 \qquad \mathsf{even}(M) = M \qquad \mathsf{odd}(M) = N$$

$$\mathsf{head}(N) = 1 \qquad \mathsf{even}(N) = N \qquad \mathsf{odd}(N) = M$$

which is a system of two equations that has the Thue-Morse sequence M (and its complement N) as its unique solution.

As we shall demonstrate, the relevance of the above non-standard stream calculus for the theory of automatic sequences lies in the following observations:

- the definition of automatic sequences by finite automata becomes a universal construction, by the finality of A^ω mentioned above;
- a stream σ is automatic iff the subcoalgebra $[\sigma]$ of the final coalgebra A^ω, obtained by repeatedly applying the operations of odd and even, is finite (much in the same way as a formal language L is regular iff the subcoalgebra it generates in the final coalgebra 2^{A^*} of all formal languages is finite);
- the size of this generated subcoalgebra gives us a well-defined notion of minimal representation for automatic sequences;

- the set of all automatic sequences is final among (a subclass of) all finitely-generated 2-automata;
- a sequence is automatic iff it can be defined by a finite system of $\{\mathsf{head}, \mathsf{even}, \mathsf{odd}\}$-stream differential equations;
- equality of automatic sequences can be proved by coinduction, using $\{\mathsf{head}, \mathsf{even}, \mathsf{odd}\}$-bisimulation relations;
- a more liberal use of non-standard stream differential equations leads to potentially interesting extensions of the set of automatic sequences, similar to the way in which context-free languages extend regular languages.

2 Streams and automata

Let A be an arbitrary set and let $2 = \{0, 1\}$. A 2-*automaton* with outputs in A is a pair $(S, \langle o, n\rangle)$ consisting of a (finite or infinite) set S of states and a pair of functions

$$\langle o, n\rangle : S \to A \times S^2$$

assigning to each state $s \in S$ its output $o(s) \in A$ and, for each input $i \in 2$, a next state $n(s)(i) \in S$. (Equivalently, 2-automata are coalgebras of the set-functor $F(X) = A \times X^2$.)

Note that every state s has two successor states, $n(s)(0)$ and $n(s)(1)$, which we shall sometimes call the 0-derivative and the 1-derivative of s, and which we often denote by

$$s_0 = n(s)(0) \qquad s_1 = n(s)(1)$$

The notion of derivative can be generalised to binary words as usual.

Definition 1. *For a state $s \in S$ in a 2-automaton $(S, \langle o, n\rangle)$ and for $w \in 2^*$, we define the w-derivative s_w of s by*

$$s_\epsilon = s \qquad s_{w\cdot b} = n(s_w)(b)$$

where ϵ is the empty word and $b \in 2$. □

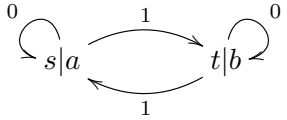
As usual, automata can be conveniently represented by pictures, in which we denote outputs by

$$s|a \iff o(s) = a$$

and successor states by labeled arrows:

$$s \xrightarrow{0} s_0 \qquad s \xrightarrow{1} s_1$$

Using these conventions, here is an example of a 2-automaton with outputs in $A = \{a, b\}$:



This picture represents an automaton with $S = \{s, t\}$; with outputs $o(s) = a$ and $o(t) = b$; and with derivatives (that is, successor states) $s_0 = s = t_1$ and $s_1 = t = t_0$.

As we already saw in the introduction, the set of streams over A forms a 2-automaton

$$\langle \mathsf{head}, (\mathsf{even}, \mathsf{odd})\rangle : A^\omega \to A \times (A^\omega)^2 \tag{1}$$

given by $\mathsf{head}(\sigma) = \sigma(0) \in A$ and by

$$\mathsf{even}(\sigma) = (\sigma(0), \sigma(2), \sigma(4), \ldots) \qquad \mathsf{odd}(\sigma) = (\sigma(1), \sigma(3), \sigma(5), \ldots)$$

In line with the conventions for 2-automata introduced above, we shall often write

$$\sigma_0 = \mathsf{even}(\sigma) \qquad \sigma_1 = \mathsf{odd}(\sigma)$$

Applying Definition 1 to the automaton of streams then gives us word derivatives of streams, which will play a crucial role later. For instance,

$$\sigma_{011} = \mathsf{odd} \circ \mathsf{odd} \circ \mathsf{even}(\sigma) = (\sigma(6), \sigma(14), \sigma(22), \ldots)$$

Because $\mathsf{even}(\sigma) = (\sigma(0), \sigma(2), \sigma(4), \ldots)$, the output values of σ and its 0-derivative σ_0 are always the same:

$$\mathsf{head}(\sigma) = \sigma(0) = \mathsf{head}(\mathsf{even}(\sigma)) = \mathsf{head}(\sigma_0)$$

Definition 2 (zero-consistency). *More generally, we call a* 2*-automaton* $(S, \langle o, n\rangle)$ zero-consistent *if* $o(n(s)(0)) = o(s)$. □

For another interesting (but not zero-consistent) 2-automaton, we consider the set

$$A^{2^*} = \{L \mid L : 2^* \to A\}$$

of all A-weighted languages over 2, together with output and transition functions

$$\langle o, n\rangle : A^{2^*} \to A \times (A^{2^*})^2 \tag{2}$$

given, for any $L : 2^* \to A$, by $o(L) = L(\epsilon)$ and by $n(L)(i) = L_i$ with

$$L_i(w) = L(i \cdot w)$$

for $i \in 2$ and $w \in 2^*$. (As an aside, we note that A-weighted languages over 2 can also be viewed as binary trees with (node) labels in A.)

3 Finality

We shall show that the automaton of all A-weighted languages is final in the family (the category) of all 2-automata. This observation is an instance of a well-known, more general characterisation of the final coalgebra for (Mealy and) Moore machines with arbitrary inputs and outputs. Next we will show that the automaton of all streams is final in the family (the subcategory) of all *zero-consistent* 2-automata. This fact is much less well-known and was recently proved in [KR10], building on earlier work by Rosu [Ros00]. The main contribution of the present paper, in Section 4, will be a new definition and characterisation of automatic sequences, based on the finality of the 2-automaton of streams.

We begin by recalling some basic notions from universal coalgebra [Rut00]. A *homomorphism* from a 2-automaton $(S, \langle o_S, t_S\rangle)$ to a 2-automaton $(T, \langle o_T, n_T\rangle)$ is a function $f : S \to T$ such that for all $s \in S$,

$$o_T(f(s)) = o_S(s) \text{ and } f(s)_0 = f(s_0), \quad f(s)_1 = f(s_1)$$

Equivalently, f is a homomorphism if it makes the following diagram commute:

$$\begin{array}{ccc} S & \xrightarrow{\quad f \quad} & T \\ {\scriptstyle \langle o_S, n_S \rangle}\downarrow & & \downarrow{\scriptstyle \langle o_T, n_T \rangle} \\ A \times S^2 & \xrightarrow[1_A \times f^2]{} & A \times T^2 \end{array}$$

Here 1_A is the identity map on A and $f^2 : S^2 \to T^2$ sends (s_0, s_1) to $(f(s_0), f(s_1))$.

Using the notion of homomorphism, we can formulate the following universal property of the automaton of languages.

Theorem 3 (finality of weighted languages). *The* 2*-automaton of* A*-weighted languages defined in equation (2) is final in the family of* all 2*-automata. That is, for every* 2*-automaton* $(S, \langle o_S, n_S \rangle)$ *there exists a unique homomorphism into the automaton* $(A^{2^*}, \langle o, n \rangle)$*:*

$$\begin{array}{ccc} S & \overset{\exists! \, l}{- - \to} & A^{2^*} \\ {\scriptstyle \langle o_S, n_S \rangle}\downarrow & & \downarrow{\scriptstyle \langle o, n \rangle} \\ A \times S^2 & \xrightarrow[1_A \times l^2]{} & A \times (A^{2^*})^2 \end{array}$$

Proof. We define $l : S \to A^{2^*}$, for $s \in S$ and $w \in 2^*$, by

$$l(s)(w) = o_S(s_w)$$

One can easily show that this turns l into a homomorphism and that this is the only way to do it. □

Next we turn to the, for the purposes of this paper more important automaton of streams. First we define the binary representation of the natural numbers

$$\mathsf{bin} : \mathbb{N} \to 2^*$$

by $\mathsf{bin}(0) = \epsilon$, the empty word, and further, as usual, by

$$\mathsf{bin}(1) = 1 \quad \mathsf{bin}(2) = 01 \quad \mathsf{bin}(3) = 11 \quad \mathsf{bin}(4) = 001$$

and so on (least significant digit first). We have the following useful property.

Lemma 4. *For* $\sigma \in A^\omega$ *and* $n \geq 0$*:* $\sigma(n) = \sigma_{\mathsf{bin}(n)}(0)$. □

For instance, $\sigma(6) = \sigma_{\mathsf{bin}(6)}(0) = \sigma_{011}(0) = (\mathsf{odd} \circ \mathsf{odd} \circ \mathsf{even}(\sigma))(0)$.

We just saw that the automaton of all weighted languages is final among all 2-automata. Next we show that the automaton of streams if final among *all zero-consistent* 2-automata.

Theorem 5 (finality of streams). *For every zero-consistent* 2*-automaton* $(S, \langle o_S, n_S \rangle)$ *there exists a unique homomorphism into the automaton* $(A^\omega, \langle \mathsf{head}, (\mathsf{even}, \mathsf{odd}) \rangle)$*:*

$$\begin{array}{ccc} S & \overset{\exists! \, h}{- - \to} & A^\omega \\ {\scriptstyle \langle o_S, n_S \rangle}\downarrow & & \downarrow{\scriptstyle \langle \mathsf{head}, (\mathsf{even}, \mathsf{odd}) \rangle} \\ A \times S^2 & \xrightarrow[1_A \times h^2]{} & A \times (A^\omega)^2 \end{array}$$

Proof. We define $h : S \to A^\omega$, for $s \in S$ and $n \geq 0$, by

$$h(s)(n) = o_S(s_{\mathsf{bin}(n)})$$

Using Lemma 4, one can show that this is the only possible definition ensuring that h is a homomorphism. □

4 Automatic sequences

We can now use the finality of the automaton of streams to give our definition of automatic sequence. We need another definition first.

Definition 6 (streams and automata). *We say that a stream $\sigma \in A^\omega$ is* generated *by a state s of a (finite or infinite) zero-consistent 2-automaton $(S, \langle o, n\rangle)$ if $\sigma = h(s)$, where $h : S \to A^\omega$ is the (by finality unique) homomorphism of Theorem 5.* □

Definition 7 (automatic). *We call a stream $\sigma \in A^\omega$* automatic *if it is generated by a* finite *zero-consistent 2-automaton.* □

We can also characterise automatic sequences in terms of *subautomata*, which we introduce next. For a state $s \in S$ in a 2-automaton $(S, \langle o_S, n_S\rangle)$, we define the *subautomaton induced by s* as the smallest set $[s] \subseteq S$ such that:

$$s \in [s] \quad \text{and} \qquad \forall t \in [s] : \; t_0 \in [s] \quad \text{and} \quad t_1 \in [s]$$

and with outputs and transitions as in S.

Theorem 8 (automatic sequences and subautomata). *For a stream $\sigma \in A^\omega$, the following are equivalent:*

(i) σ is automatic.
(ii) The subautomaton $[\sigma] \subseteq A^\omega$ induced by σ is finite*.*
(iii) σ has only finitely many derivatives.

Proof. The equivalence of (i) and (ii) follows from elementary universal coalgebra [Rut00]: (ii) implies (i) because the inclusion of a subautomaton into a bigger automaton is always a homomorphism; and (i) implies (ii) because the image of a subautomaton under a homomorphism is always a subautomaton or, more specifically: $h([s]) = [h(s)]$. The equivalence of (ii) and (iii) follows from the definition of subautomaton. □

By this theorem, we have a very precise correspondence between automatic sequences and regular languages, namely:

$$\frac{\text{automatic sequences}}{\text{all streams}} \;=\; \frac{\text{regular languages}}{\text{all languages}}$$

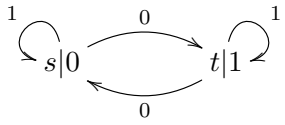
Let us explain. According to Theorem 8 above, a stream σ (over A) is automatic if and only if the subautomaton $[\sigma]$ induced by σ in the final coalgebra A^ω of all streams is finite or, equivalently, if σ has only finitely many derivatives. There is the following similar fact for languages [Con71,Rut98]: a language L (over an alphabet A) is regular if and only if the subautomaton $[L]$ induced by L in the final coalgebra 2^{A^*} of all languages is finite or, equivalently, if L has only finitely many input derivatives.

Discussion

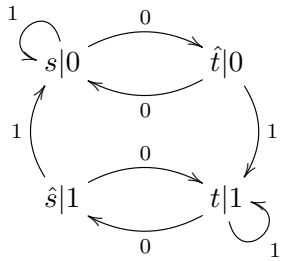
In the remainder of this section, we briefly discuss how our definition of automaticity is related to the existing classical notion(s) in the literature. None of this will play a role in the following sections.

Definition 7 is (almost) equivalent to the following definition [Fog02, page 13]: an infinite sequence (stream) is 2-automatic if it is *generated by a 2-automaton in reverse reading* (reading the least significant digit of the binary representation of the natural numbers first). Without spelling out the details of this latter definition, it is almost the same as ours.

We say *almost* because in the classical definition, 2-automata are not required to be zero-consistent. So our present definition might seem more restrictive but it is not. We illustrate this by means of a simple example, omitting the not particularly instructive proof that this holds in general. Let us consider the following example of a 2-automaton with outputs in $A = \{0, 1\}$:



We note that it is not zero-consistent since the output of s is 0 and the output of $s_0 = t$ is 1. However, it can be transformed into the following four state zero-consistent automaton:



which is equivalent to the original automaton in the following sense: One can easily show that the stream generated in reverse reading by this latter automaton (starting in $\hat{s}$) is the same as the stream generated in reverse reading by the original automaton (starting in s).

Then there exists yet another classical definition of automaticity, in which a stream is generated while reading the most significant digit of the binary representation of the natural numbers first. Such a stream is said to be generated in *direct reading*. It has been shown ([Fog02, Proposition 1.3.4.], [AS03, Theorem 5.2.3]) that automaticity in reverse reading and in direct reading are equivalent.

Finally, our Theorem 8 above is also known, be it somewhat implicitly, in the literature, where our (ii) is phrased in terms of 2-kernels; see [Fog02, Proposition 1.3.3.] and [AS03, Theorem 6.6.2] for details.

5 Non-automatic sequences

Theorem 8 gives us a convenient criterium to decide whether a stream is automatic or not.

For a first example, let us look at the characteristic stream of powers of 2:

$$\tau = (0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, \ldots)$$

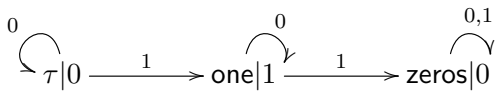
Using elementary stream calculus [Rut05], with $X = (0, 1, 0, 0, \ldots)$, convolution product as multiplication and element-wise addition as sum, we can write τ as

$$\begin{aligned}\tau &= X^{2^0} + X^{2^1} + X^{2^2} + X^{2^3} + \cdots \\ &= X + X^2 + X^4 + X^8 + \cdots\end{aligned}$$

(Equivalently, this can be viewed as a power series representation of τ in the formal variable X.) Computing the (repeated) derivatives of τ gives

$$\tau_0 = \tau \quad \tau_1 = \mathsf{one} \quad \mathsf{one}_0 = \mathsf{one} \quad \mathsf{one}_1 = \mathsf{zeros} \quad \mathsf{zeros}_0 = \mathsf{zeros} = \mathsf{zeros}_1$$

where $\mathsf{one} = (1, 0, 0, \ldots)$ and $\mathsf{zeros} = (0, 0, 0, \ldots)$. Thus we see that τ is generated by the following finite automaton



By Theorem 8, we conclude that τ is automatic.

For a second example, we consider the characteristic sequence of squares

$$\rho = (1, 1, 0, 0, 1, 0, 0, 0, 0, 1, \ldots)$$

or, equivalently,

$$\begin{aligned}\rho &= X^{0^2} + X^{1^2} + X^{2^2} + X^{3^2} + \cdots \\ &= 1 + X + X^4 + X^9 + \cdots\end{aligned}$$

In order to decide whether ρ is automatic or not, we investigate its derivatives. Since

$$\rho_0 = 1 + X^2 + X^8 + X^{18} + X^{32} + X^{50} + X^{72} + \cdots$$

it follows that $\rho_{00} = \rho$ and $\rho_{01} = \mathsf{zeros}$. Next we compute

$$\begin{aligned}\rho_1 &= 1 + X^4 + X^{12} + X^{24} + X^{40} + \cdots \\ \rho_{10} &= 1 + X^2 + X^6 + X^{12} + X^{20} + \cdots \\ \rho_{100} &= 1 + X + X^3 + X^6 + X^{10} + X^{15} + \cdots\end{aligned}$$

where we note that the latter stream is pleasantly regular:

$$\rho_{100} = X^0 + X^{0+1} + X^{0+1+2} + X^{0+1+2+3} + X^{0+1+2+3+4} + X^{0+1+2+3+4+5} + \cdots$$

Next we can easily prove by induction that the first non-zero coefficient of X of the subsequent zero derivatives satisfies, for all $n \geq 1$,

$$\rho_{10^{n+1}} = 1 + X^{2^n - 1} + \cdots$$

Thus all these streams are different. By Theorem 8, we have that ρ is *not* automatic. The reader is invited to compare this proof with that of [AS03, page 167], which is somewhat ad hoc and which is based on the pumping lemma for formal languages.

6 Coinduction (definitions)

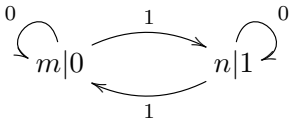
In coalgebra, the property of being final gives rise to both a method for *defining* and a method for *proving* equality (of the elements of the final coalgebra). Such definitions and proofs are often called *coinductive* or: *by coinduction*. In the present section, we shall sketch how coinductive definitions looks like for (automatic) streams. The next section will deal with coinductive proofs.

Coinductive definitions of streams are given by $\{\mathsf{head}, \mathsf{even}, \mathsf{odd}\}$-stream differential equations. As an example, we consider the Thue-Morse sequence and its inverse, which we already saw in the introduction. Let the streams $M, N \in 2^\omega$ be defined by the following equations:

$$\begin{array}{lll} \mathsf{head}(M) = 0 & \mathsf{even}(M) = M & \mathsf{odd}(M) = N \\ \mathsf{head}(N) = 1 & \mathsf{even}(N) = N & \mathsf{odd}(N) = M \end{array} \tag{3}$$

We mentioned in the introduction that this system of two differential equations has the Thue-Morse sequence M and its complement N as its unique solution. But how do we know that the system has a solution at all and that this solution is moreover unique?

The affirmative answer is given by finality. We use the differential equations to define a zero-consistent 2-automaton S as follows: let $S = \{m, n\}$ and let outputs (in $A = 2$) and transitions be given by the following picture:



By finality Theorem 5, there exists a unique homomorphism $h : S \to 2^\omega$ which we use to define

$$M = h(m) \quad N = h(n)$$

It is now straightforward to show that the streams M and N are solutions of, that is, satisfy the differential equations above. Computing their first elements gives

$$M = (0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, \ldots)$$

$$N = (1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, \ldots)$$

Moreover, because every pair of solutions of our differential equations will give rise to a homomorphism from S to 2^ω, the unicity of h implies that M and N are the unique solutions. Finally, because S is finite both M and N are automatic.

More generally, we call a finite system of $\{\mathsf{head}, \mathsf{even}, \mathsf{odd}\}$-stream differential equations *simple* if it is of the following form:

$$\mathsf{head}(x_1) = a_1 \quad \mathsf{even}(x_1) = y_1 \quad \mathsf{odd}(x_1) = z_1$$

$$\ldots$$

$$\mathsf{head}(x_n) = a_n \quad \mathsf{even}(x_n) = y_n \quad \mathsf{odd}(x_n) = z_n$$

where $X = \{x_1, \ldots, x_n\}$ is the set of unknowns, for some $n \geq 1$; and where $a_i \in A$, $y_i \in X$ and $z_i \in X$, for all $1 \leq i \leq n$. We call such a simple system *zero-consistent* if $\mathsf{head}(y_i) = \mathsf{head}(x_i)$, for all $1 \leq i \leq n$.

Theorem 9 (simple systems of differential equations). *A stream $\sigma \in A^\omega$ is automatic iff it is the solution of a simple zero-consistent system of* $\{\mathsf{head}, \mathsf{even}, \mathsf{odd}\}$*-stream differential equations.*

Proof. The proof essentially consists of the observation that simple zero-consistent systems of $\{\mathsf{head}, \mathsf{even}, \mathsf{odd}\}$-stream differential equations are in one-to-one correspondence to finite zero-consistent 2-automata. □

We refer the reader to [KR10] for a description of some more general well-defined classes of $\{\mathsf{head}, \mathsf{even}, \mathsf{odd}\}$-stream differential equations.

Next we illustrate that one can use $\{\mathsf{head}, \mathsf{even}, \mathsf{odd}\}$-stream differential equations also to define *functions* on streams. For instance, we can define the function

$$\mathsf{inv} : 2^\omega \to 2^\omega$$

which replaces 0's by 1's and 1's by 0's, by the following equations:

$$\mathsf{inv}(\sigma)(0) = 1 - \sigma(0) \qquad \mathsf{inv}(\sigma)_0 = \mathsf{inv}(\sigma_0) \qquad \mathsf{inv}(\sigma)_1 = \mathsf{inv}(\sigma_1) \tag{4}$$

where we now write $-(0)$ for $\mathsf{head}(-)$ and where we use subscripts 0 and 1 to denote the operations of even and odd. The example is simple enough to see at once that there exists precisely one function satisfying these equations. Formally, a proof can be based as before on the finality of streams: we define an (infinite) 2-automaton $(T, \langle o, n\rangle)$ by

$$T = 2^\omega$$

with observations and transitions

$$o(\sigma) = 1 - \sigma(0) \quad n(\sigma)(0) = \sigma_0 \quad n(\sigma)(1) = \sigma_1$$

We note that this automaton is zero-consistent: for all $\sigma \in 2^*$,

$$(n(\sigma)(0))(0) = (\sigma_0)(0) = (\mathsf{even}(\sigma))(0) = \sigma(0)$$

Hence there exists, by finality, a unique homomorphism $h : T \to 2^\omega$, which we use to define the function $\mathsf{inv} = h$. It is easy to check that this function is a solution of the differential equation that we started with, and that it is the only solution.

Although the defining automaton T for inv is infinite, there still is a connection with automaticity. Namely, if σ is an automatic stream then so is $\mathsf{inv}(\sigma)$. Using Theorem 8, a proof is easy: if σ is automatic, it has only finitely many (0- and 1-) derivatives. This implies that (both c_σ and) $\mathsf{inv}(\sigma)$ has only finitely many derivatives. Which implies, again by Theorem 8, that $\mathsf{inv}(\sigma)$ is automatic.

Here is an example where the right-hand sides of the equations are no longer simple but may involve the use of stream functions. Recalling the (trivial) definitions of the streams $\mathsf{zeros} = (0, 0, 0, \ldots)$ and $\mathsf{ones} = (1, 1, 1, \ldots)$ from the introduction:

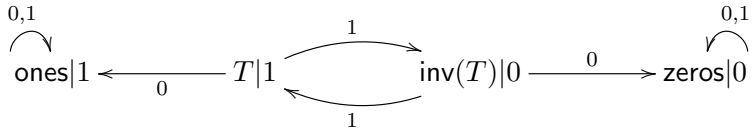
$$\mathsf{zeros}(0) = 0 \qquad \mathsf{zeros}_0 = \mathsf{zeros} \qquad \mathsf{zeros}_1 = \mathsf{zeros}$$

$$\mathsf{ones}(0) = 1 \qquad \mathsf{ones}_0 = \mathsf{ones} \qquad \mathsf{ones}_1 = \mathsf{ones}$$

and using the function inv introduced above, we consider the following differential equation:

$$T(0) = 1 \qquad T_0 = \mathsf{ones} \qquad T_1 = \mathsf{inv}(T)$$

which defines the so-called *Toeplitz* sequence. The equation can be shown to have a unique solution in a similar fashion to the examples above, by deriving from the defining differential equations for T, ones and inv a 2-automaton of the following shape:



(Here we have used the facts that $\mathsf{inv}(\mathit{ones}) = \mathsf{zeros}$ and that $\mathsf{inv}(\mathsf{inv}(\sigma)) = \sigma$, for all σ. These facts are elementary and can be formally proved by the coinduction proof technique of Section 7.) We note hat T is automatic, since the automaton above is finite.

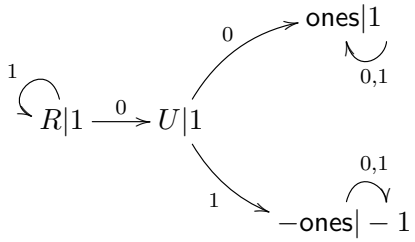
Another famous example of an automatic sequence (over the set $A = \{1, -1\}$) is the Regular Paperfolding Sequence R, which we define here by the following differential equation:

$$R(0) = 1 \qquad R_0 = U \qquad R_1 = R$$

Here U is a stream defined by

$$U(0) = 1 \qquad U_0 = \mathsf{ones} \qquad U_1 = -\mathsf{ones}$$

and minus is defined as expected. Again the existence of a unique solution of the equations for R and U above can be proved by finality, by the construction of an automaton of the shape



This is a zero-consistent finite 2-automaton, so both R and U are automatic.

Here is yet another example of a coinductive definition of a *function* on streams. Assuming that A has a binary operation $\cdot$ of multiplication, we define the so-called *Hadamard* product $\sigma \odot \tau$, for all streams $\sigma, \tau \in A^\omega$, by

$$(\sigma \odot \tau)(0) = \sigma(0) \cdot \tau(0) \qquad (\sigma \odot \tau)_0 = \sigma_0 \odot \tau_0 \qquad (\sigma \odot \tau)_1 = \sigma_1 \odot \tau_1$$

The product stream $\sigma \odot \tau$ consists of the elementwise products in A of the elements of σ and τ. We note that the Hadamard product preserves automaticity: if both σ and τ have only finitely many derivatives then so does $\sigma \odot \tau$.

7 Coinduction (proofs)

A *bisimulation* between two zero-consistent 2-automata $(S, \langle o_S, t_S\rangle)$ and $(T, \langle o_T, n_T\rangle)$ is a relation $R \subseteq S \times T$ such that for all $(s,t) \in S \times T$

$$(s,t) \in R \Rightarrow \begin{cases} o_S(s) = o_T(t) & \text{and} \\ (s_i, t_i) \in R & i = 1,2 \end{cases}$$

(where, as before, $s_i = n_S(s)(i)$ and $t_i = n_T(t)(i)$).

Equivalently, $R \subseteq S \times T$ is a bisimulation if it can be turned into a 2-automaton $(R, \langle o, n\rangle)$ such that the projections $l : R \to S$ and $r : R \to T$ (which are defined by $l((s,t)) = s$ and $r((s,t)) = t$) are homomorphisms:

$$\begin{array}{ccccc} S & \xleftarrow{\ l\ } & R & \xrightarrow{\ r\ } & T \\ {\scriptstyle \langle o_S, n_S\rangle}\downarrow & & \downarrow{\scriptstyle \langle o, n\rangle} & & \downarrow{\scriptstyle \langle o_T, n_T\rangle} \\ A \times S^2 & \xleftarrow[1_A \times l^2]{} & A \times R^2 & \xrightarrow[1_A \times r^2]{} & A \times T^2 \end{array}$$

There is the following powerful proof principle for streams.

Theorem 10 (coinduction proof principle). *For any bisimulation relation $R \subseteq A^\omega \times A^\omega$ on the automaton $(A^\omega, \langle \mathsf{head}, (\mathsf{even}, \mathsf{odd})\rangle)$ of streams, we have, for all $\sigma, \tau \in A^\omega$,*

$$\text{if } (\sigma, \tau) \in R \text{ then } \sigma = \tau$$

Proof. By the finality of A^ω and the definition of bisimulation, the projections $l, r : R \to A^\omega$ are equal. As a consequence, $\sigma = \tau$ for all $(\sigma, \tau) \in R$. □

Thus in order to show the equality of two streams, it suffices to construct a bisimulation relation that contains the pair of those streams. For a first example, we consider the function $\mathsf{zip} : A^\omega \times A^\omega \to A^\omega$ defined, for all $\sigma, \tau \in A^\omega$, by the following stream differential equation:

$$\mathsf{zip}(\sigma,\tau)(0) = \sigma(0) \quad \mathsf{zip}(\sigma,\tau)_0 = \sigma \quad \mathsf{zip}(\sigma,\tau)_1 = \tau$$

As the name suggests, this function satisfies

$$\mathsf{zip}(\sigma,\tau) = (\sigma(0), \tau(0), \sigma(1), \tau(1), \ldots)$$

We have the following trivial identity, for all $\sigma \in A^\omega$:

$$\mathsf{zip}(\sigma_0, \sigma_1) = \sigma \tag{5}$$

It can be viewed as a kind of *fundamental theorem* of $\{\mathsf{head}, \mathsf{even}, \mathsf{odd}\}$-stream calculus, in that it enables one to compute σ from its derivatives.

Equality (5) follows by coinduction Theorem 10 from the fact that the following relation $R \subseteq A^\omega \times A^\omega$ is a bisimulation:

$$R = \{(\mathsf{zip}(\sigma_0, \sigma_1),\ \sigma) \mid \sigma \in A^\omega\} \cup \{(\sigma,\ \sigma) \mid \sigma \in A^\omega\}$$

For another example of coinduction, we prove the following equality:

$$M = \mathsf{zip}(M, \mathsf{inv}(M))$$

where M is the Thue-Morse sequence defined in equation (3) and where inv is defined in equation (4). For a proof by coinduction, we define a relation $U \subseteq 2^\omega \times 2^\omega$ consisting of the following four pairs:

$$U = \{\, (M, \mathsf{zip}(M, \mathsf{inv}(M))),\ (M, M),\ (M, \mathsf{inv}(N)),\ (N, \mathsf{inv}(M)) \,\}$$

It is easy to check that U is a bisimulation. Thus $M = \mathsf{zip}(M, \mathsf{inv}(M))$ follows by coinduction Theorem 10.

8 Discussion

Conclusion: We have given a new perspective on automatic sequences, by defining them with a universal construction from coalgebra: finality. We have shown that our new definition is equivalent to (one of) the classical definition(s), thereby illustrating that the latter is in a precise sense universal. (Although we have talked about 2-automatic sequences only, everything can be easily adapted for k-automatic sequences, for arbitrary k.) Using our new definition, we have shown that a stream is automatic if and only if it has only finitely many (even and odd) derivatives (Theorem 8). This characterisation offers an equivalent but convenient alternative to the more classical use of 2-kernels, as was illustrated by the examples in Section 5.

We also saw that a stream σ has only finitely many derivatives if and only if it generates a finite subautomaton $[\sigma] \subseteq A^\omega$. As a consequence, we can take the size of this subautomaton $[\sigma]$ as a well-defined notion of minimal representation for automatic sequences.

Further we have shown that a stream is automatic if and only if it can be defined by means of a simple zero-consistent system of $\{\mathsf{head}, \mathsf{even}, \mathsf{odd}\}$-stream differential equations (Theorem 9). As we have seen, such equations can be used also to define functions on streams, allowing one to give easy proofs that certain functions preserve automaticity.

Related work: Our knowledge about automatic sequences is based on the afore mentioned [Fog02] and [AS03], which offer a rich body of results on automatic sequences and their applications. It goes without saying that in the present paper, we have dealt with only a very small part thereof. We have explained above what might be seen as our contribution to the field of automatic sequences: a new, universal perspective on the definition of automatic sequence and, related to that, a characterisation of automaticity in terms of (even and odd) derivatives.

All the technical results of our paper are already contained in our earlier [KR10], or have been derived from there using some elementary universal coalgebra [Rut00]. What is really new is the observation that being generated by a finite 2-automaton (Definition 6) is equivalent to (the classical definition of) being 2-automatic. This observation is in itself trivial but making it gave us a kind of aha-erlebnis. And, once made, it opened the way of applying our earlier results.

Apparently, some of these insights were already in the air. After a recent presentation of our work at the COIN (Coalgebra in the Netherlands) seminar, we were told of ongoing research by Endrullis, Grabmayer, Hendriks, Klop and Moss [EGH$^+$11], in which some related facts have been independently established (though not in a coalgebraic framework).

Further research: We see several perspectives for further research. It would be interesting to investigate more liberal formats of stream differential equations than the *simple* systems we have considered ([KR10] contains already some first attempts). This could lead to potentially interesting extensions of the set of automatic sequences, similar to the way in which context-free languages extend regular languages.

Given the parallel between regular languages and automatic sequences, as sketched in Section 4, we intend to investigate possible definitions of what could be called languages of *automatic expressions*, and a corresponding theorem in the style of Kleene. Clearly the use of the operation of zip offers a way to translate $\{\mathsf{head}, \mathsf{even}, \mathsf{odd}\}$-stream differential equations into closed expressions, but we would also be interested in operations such as (a variant on) the Kleene star or maybe some fixed point operator.

Finally, given the well-established correspondence between coalgebra and (modal) logic, it might be worthwhile to design new logics for reasoning about automatic sequences and their properties.

References

[AS03] J.-P. Allouche and J. Shallit. *Automatic Sequences: Theory, Applications, Generalizations.* Cambridge University Press, 2003.

[Con71] J.H. Conway. *Regular algebra and finite machines.* Chapman and Hall, 1971.

[EGH+11] J. Endrullis, C. Grabmayer, D. Hendriks, J.W. Klop, and Larry Moss. Zip-specifications and automatic sequences. 2011. Forthcoming.

[Fog02] N. Pytheas Fogg. *Substitutions in dynamics, arithmetics and combinatorics*, volume 1794 of *Lecture Notes in Math.* Springer, Berlin, 2002. Edited by Valérie Berthé, Sébastien Ferenczi, Christian Mauduit and Anne Siegel.

[KR10] C. Kupke and J.J.M.M. Rutten. Complete sets of cooperations. *Information and Computation*, 208(12):1398–1420, 2010.

[Ros00] G. Rosu. *Hidden Logic.* PhD thesis, University of California at San Diego, 2000.

[Rut98] J.J.M.M. Rutten. Automata and coinduction (an exercise in coalgebra). In D. Sangiorgi and R. de Simone, editors, *Proceedings of CONCUR'98*, volume 1466 of *LNCS*, pages 194–218, 1998.

[Rut00] J.J.M.M. Rutten. Universal coalgebra: a theory of systems. *Theoretical Computer Science*, 249(1):3–80, 2000. Fundamental Study.

[Rut05] J.J.M.M. Rutten. A coinductive calculus of streams. *Mathematical Structures in Computer Science*, 15:93–147, 2005.