Well-definedness of streams by termination

Please check the document version of this publication:

• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

Download date: 04. Oct. 2023

Well-Definedness of Streams by Termination

Hans Zantema[1,2]

[1] Department of Computer Science, TU Eindhoven, P.O. Box 513,
5600 MB Eindhoven, The Netherlands
H.Zantema@tue.nl

[2] Institute for Computing and Information Sciences, Radboud University
Nijmegen, P.O. Box 9010, 6500 GL Nijmegen, The Netherlands

Abstract. Streams are infinite sequences over a given data type. A stream specification is a set of equations intended to define a stream. We propose a transformation from such a stream specification to a TRS in such a way that termination of the resulting TRS implies that the stream specification admits a unique solution. As a consequence, proving such well-definedness of several interesting stream specifications can be done fully automatically using present powerful tools for proving TRS termination.

1 Introduction

Streams are among the simplest data types in which the objects are infinite. We consider streams to be maps from the natural numbers to some data type D. The basic constructor for streams is the operator ':' mapping a data element d and a stream s to a new stream $d : s$ by putting d in front of s. Using this operator we can define streams by equations. For instance, the stream zeros only consisting of 0's can be defined by the single equation zeros $= 0$: zeros. More complicated streams are defined using stream functions. For instance, the boolean *Fibonacci stream* Fib is defined[1] to be the fixpoint of the function f defined by

$$f(0 : \sigma) = 0 : 1 : f(\sigma), \quad f(1 : \sigma) = 0 : f(\sigma).$$

It turns out that Fib $= 0 : c$ for the stream c defined by $c = 1 : f(c)$. Although these stream definitions are extremely simple, the resulting streams are typically non-periodic and have remarkable properties. For instance, one can make a *turtle visualization* (see also http://www.win.tue.nl/~hzantema/str.html) as follows. Choose an initial drawing direction and traverse the elements of the stream Fib as follows: if the symbol 0 is read then the drawing direction is moved 30 degrees to the right; if the symbol 1 is read then the drawing direction is moved 150 degrees to the left. In both cases after doing so a line of unit length is drawn. Then after 200.000 steps the following picture is obtained.

---

[1] In [1] it is called infinite Fibonacci word. It can also be defined as the limit of the strings ϕ_i where $\phi_1 = 1$, $\phi_2 = 0$, $\phi_{i+2} = \phi_{i+1}\phi_i$ for $i \geq 1$, showing the relationship with Fibonacci numbers.
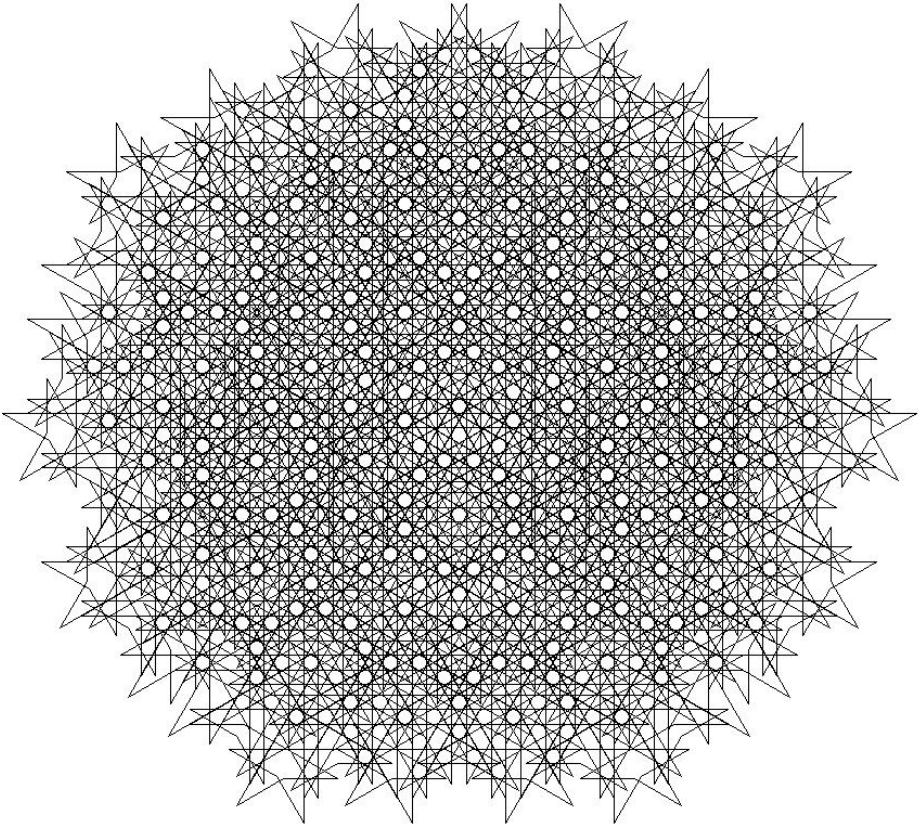
Streams have been studied extensively, e.g. in [1]. In this paper we consider stream specifications consisting of a set of equations like above we did for the Fibonacci stream. We address the most fundamental question one can think of: does such a set of equations admits a unique solution as constants and functions on streams? This is not always the case. For instance, every f mapping $x : \sigma$ to $x : c$ for any constant stream c satisfies the stream specification

$$f(x : \sigma) = x : g(f(\sigma)), \quad g(x : \sigma) = \sigma,$$

where g is the tail function removing the first element of the stream.

Intuitively this notion of well-definedness is closely related to termination of the process of unfolding definitions. The past ten years showed up a remarkable progress in techniques and implementations for proving termination of rewrite systems [2,5,9]. One of the objectives of this paper is to exploit this power for proving well-definedness of stream specifications. In our approach we introduce fresh operators head and tail intended to observe streams. We present a transformation of the specification to its *observational variant*. This is a TRS mimicking the stream specification in such a way that head or tail applied on any stream constant or stream function can always be rewritten. This transformation is

straightforward and easy to implement; an implementation for boolean stream specifications, both in Windows an Linux, together with several examples, is found in http://www.win.tue.nl/~hzantema/str.zip.

The main result of this paper states that if the observational variant of a specification is terminating, then the specification admits a unique solution. It turns out that for several interesting cases termination of the observational variant of a specification can be proved by termination tools like AProVE [4] or TTT2 [7]. This provides a new technique to prove well-definedness of stream specifications fully automatically, applying for cases where earlier approaches fail. Our main result appears in two variants:

- a variant restricting to ground terms for general stream specifications (Theorem 1), and
- a variant generalizing to all streams for stream specifications not depending on particular data elements (Theorem 2).

By an example we show that the approach does not work for general stream specifications and functions applied on all streams. Moreover, we show that our technique is not complete: the fixpoint definition of the Fibonacci stream as we just gave is a well-defined stream specification for which the observational variant is non-terminating.

Proving well-definedness in stream specification is closely related to proving equality of streams. A standard approach for this is co-induction [11]: two streams or stream functions are equal if a bisimulation can be found between them. Finding such an arbitrary bisimulation is a hard problem in the general setting, but restricting to circular co-induction [6] finding this automatically is tractable. A strong tool doing so is Circ [8]. The tool Circ focuses on proving equality, but proving well-definedness of a function f can also be proved by equality as long as the equations for f are orthogonal: take a copy f' of f with the same equations, and prove $f = f'$. For many examples this works well, but there are also small stream specifications for which our approach succeeds in proving well-definedness and Circ fails. Conversely our approach can be used to prove equality of two streams: if one stream satisfies the specification of the other one, and this specification is well-defined, then the streams are equal. The input format of Circ differs from what we call stream specifications: head and tail are already building blocks and the Circ input is essentially the same as what we call the observational variant.

Another closely related topic is *productivity* of stream specifications, as studied by [3]. Productive stream specifications are always well-defined. Conversely we will give an example (Example 4) of a stream specification that is well-defined, but not productive. Our format of stream specifications is strongly inspired by [3]. In [3] a technique is developed for establishing productivity fully automatically for a restricted class of stream specifications. In particular, only a very mild type of nesting in the right hand sides of the rule is allowed. Our technique typically applies where these restrictions do not hold.

Both stream equality [10] and productivity [12] have been proved to be Π_2^0-complete, hence undecidable. By similar Turing machine construction the same is expected to hold for stream well-definedness.

This paper is structured as follows. In Section 2 we present the basics of stream specifications and their models. In Section 3 we define the transformation of a stream specification to its observational variant. In Section 4 we present and prove the main theorem: if the observational variant is terminating then restricted to ground terms the specification has a unique model. In Section 5 we show that this restriction to ground terms may be removed in case the stream specification is data independent: left hand sides of rules do not contain data values. In Section 6 we discuss fixpoints and prove incompleteness. We conclude in Section 7.

2   Streams: Specifications and Models

In stream specifications we have two sorts: s (stream) and d (data). We assume the set D of data elements to consist of the unique normal forms of ground terms over some signature Σ_d with respect to some terminating orthogonal rewrite system R_d over Σ_d. Here all symbols of Σ_d are of type $d^n \to d$ for some $n \geq 0$. We assume a particular symbol : having type $d \times s \to s$. For giving the actual stream specification we need a set Σ_s of stream symbols, each being of type $d^n \times s^m \to s$ for $n, m \geq 0$. Now terms of sort s are defined inductively as follows:

- a variable of sort s is a term of sort s,
- if $f \in \Sigma_s$ is of type $d^n \times s^m \to s$, $u_1, \ldots, u_n$ are terms over Σ_d and $t_1, \ldots, t_m$ are terms of sort s, then $f(u_1, \ldots, u_n, t_1, \ldots, t_m)$ is a term of sort s,
- if u is a term over Σ_d and t is a term of sort s, then $u : t$ is a term of sort s.

As a notational convention variables of sort d will be denoted by x, y, terms of sort d by u, u_i, variables of sort s by σ, τ, and terms of sort s by t, t_i.

Definition 1. *A* stream specification $(\Sigma_d, \Sigma_s, R_d, R_s)$ *consists of* Σ_d, Σ_s, R_d *as given before, and a set* R_s *of rewrite rules of the shape*

$$f(u_1, \ldots, u_n, t_1, \ldots, t_m) \to t,$$

where

- *$f \in \Sigma_s$ is of type $d^n \times s^m \to s$,*
- *for every $i = 1, \ldots, n$ the term u_i is either a variable of sort d or $u_i \in D$,*
- *for every $i = 1, \ldots, m$ the term t_i is either a variable of sort s, or $t_i = x : \sigma$ where x is a variable of sort d and σ is a variable of sort s,*
- *t is any term of sort s,*
- *every ground term of sort s being in normal form with respect to R_d matches with the left hand side of exactly one rule from R_s.*

Due to these requirements $R_s \cup R_d$ is orthogonal. Sometimes we call R_s a stream specification: in that case Σ_d, Σ_s consist of the symbols of sort d, s, respectively, occurring in R_s, and $R_d = \emptyset$. Rules $\ell \to r$ in R_s are often written as $\ell = r$.

Example 1. For specifying the Thue Morse sequence the data elements are $0, 1$, and a data operation not is used. The data rewrite system R_d consists of the two rules $\mathsf{not}(0) \to 1$ and $\mathsf{not}(1) \to 0$. The rewrite system R_s consists of the rules

$$\mathsf{morse} \to 0 : \mathsf{zip}(\mathsf{inv}(\mathsf{morse}), \mathsf{tail}(\mathsf{morse})) \qquad \mathsf{tail}(x : \sigma) \to \sigma$$
$$\mathsf{inv}(x : \sigma) \to \mathsf{not}(x) : \mathsf{inv}(\sigma) \qquad \mathsf{zip}(x : \sigma, \tau) \to x : \mathsf{zip}(\tau, \sigma)$$

Definition 1 is closely related to the definition of stream specification in [3]. In fact there are two differences:

- We want to specify streams for every ground term of sort s, while in [3] there is a designated constant to be specified.
- The restriction on left hand sides is stronger than the exhaustiveness from [3]. However, by introducing fresh symbols and rules for defining these fresh symbols, every stream specification in the format of [3] can be unfolded to a stream specification in our format.

For instance, the function f in the introduction to define the Fibonacci stream does not meet our requirements since the argument $0 : \sigma$ in the left hand side $f(0 : \sigma)$ is not of the right shape. Introducing a fresh symbol g and unfolding yields

$$f(x : \sigma) = g(x, \sigma) \qquad g(0, \sigma) = 0 : 1 : f(\sigma)$$
$$g(1, \sigma) = 0 : f(\sigma)$$

satisfying our format.

Stream specifications are intended to specify streams for the constants in Σ_s, and stream functions for the other elements of Σ_s. The combination of these streams and stream functions is what we will call a *stream model*.

More precisely, a *stream* over D is a map from the natural numbers to D. Write D^ω for the set of all streams over D. In case of $D = \emptyset$ we have $D^\omega = \emptyset$; in case of $\#D = 1$ we have $\#D^\omega = 1$. So in non-degenerate cases we have $\#D \geq 2$.

It seems natural to require that stream functions in a stream model are defined on all streams. However, it turns out that several desired properties do not hold when requiring this. Therefore we allow stream functions to be defined on some set $S \subseteq D^\omega$ for which every ground term can be interpreted in S.

Definition 2. *A* stream model *is defined to consist of a set $S \subseteq D^\omega$ and a set of functions $[f]$ for every $f \in \Sigma_s$, where $[f] : D^n \times S^m \to S$ if the type of f is $d^n \times s^m \to s$.*

For a ground term u over Σ_d write $\mathbf{NF}(u)$ for its R_d-normal form. For $f \in \Sigma_d$ and $u_1, \ldots, u_n \in D$ we define $[f(u_1, \ldots, u_n)] = \mathbf{NF}(f(u_1, \ldots, u_n))$. We write $\mathcal{T}_s$

for the set of ground terms of sort s. For $t \in \mathcal{T}_s$ the stream interpretation $[t]$ in the stream model $(S, ([f])_{f \in \Sigma_s})$ is defined inductively by:

$$[f(u_1, \ldots, u_n, t_1, \ldots, t_m)] = [f]([u_1], \ldots, [u_n], [t_1], \ldots, [t_m]) \quad \text{for } f \in \Sigma_s$$
$$[u : t](0) = [u]$$
$$[u : t](i) = [t](i - 1) \qquad\qquad \text{for } i > 0$$

for all ground terms u, u_i of sort d and all ground terms t, t_i of sort s.

So in a stream model:

- every data operator is interpreted by its corresponding term constructor, after which the result is reduced to normal form,
- every stream operator f is interpreted by the given function $[f]$, and
- the operator : applied on a data element d and a stream s is interpreted by putting d on the first position and shifting every stream element of s to its next position.

Definition 3. *A stream model $(S, ([f])_{f \in \Sigma_s})$ is said to* satisfy *a stream specification $(\Sigma_d, \Sigma_s, R_d, R_s)$ if $[\ell\rho] = [r\rho]$ for every rule $\ell \to r$ in R_s and every ground substitution ρ. We also say that the specification* admits *the model.*

Now we can express the desired well-definedness of a stream specification more precisely: there is exactly one stream model $(S, ([f])_{f \in \Sigma_s})$ satisfying the stream specification for which $S = \{[t] \mid t \in \mathcal{T}_s\}$. This is not always the case: if $\#D > 1$ and R_s consists of the rule $c \to c$ there is not a unique $[c]$ since every stream satisfies the specification. Less trivial is the boolean stream specification

$$c = 0 : f(c), \qquad f(x : \sigma) = \sigma,$$

in which $[f]$ can be chosen to be the tail function and $[c]$ be any stream starting with 0, showing non-uniqueness of stream models.

3   The Observational Variant

In this paper we define a transformation Obs transforming the original TRS R_s to its *observational variant* $\mathsf{Obs}(R_s)$. The basic idea is that the streams are observed by two auxiliary operator head and tail, of which head picks the first element of the stream and tail removes the first element from the stream, and that for every $t \in \mathcal{T}_s$ of type stream both $\mathsf{head}(t)$ and $\mathsf{tail}(t)$ can be rewritten by $\mathsf{Obs}(R_s)$.

The main result of this paper is that if $\mathsf{Obs}(R_s) \cup R_d$ is terminating for a given specification $(\Sigma_d, \Sigma_s, R_d, R_s)$, then it admits a unique model $(S, ([f])_{f \in \Sigma_s})$ satisfying $S = \{[t] \mid t \in \mathcal{T}_s\}$. As a consequence, the specification uniquely defines a corresponding stream $[t]$ for every $t \in \mathcal{T}_s$.

We define $\mathsf{Obs}(R_s)$ in two steps. First we define $\mathsf{P}(R_s)$ obtained from R_s by modifying the rules as follows. By definition every rule of R_s is of the shape

$$f(u_1, \ldots, u_n, t_1, \ldots, t_m) \to t$$

where for every $i = 1, \ldots, m$ the term t_i is either a variable of sort s, or $t_i = x : \sigma$ where x is a variable of sort d and σ is a variable of sort s. In case $t_i = x : \sigma$ then in the left hand side of the rule the subterm t_i is replaced by σ, while in the right hand side of the rule every occurrence of x is replaced by $\mathsf{head}(\sigma)$ and every occurrence of σ is replaced by $\mathsf{tail}(\sigma)$.

For example, the zip rule in Example 1 will be replaced by

$$\mathsf{zip}(\sigma, \tau) \to \mathsf{head}(\sigma) : \mathsf{zip}(\tau, \mathsf{tail}(\sigma)).$$

Now we are ready to define Obs.

Definition 4. *Let $(\Sigma_d, \Sigma_s, R_d, R_s)$ be a stream specification. Let $\mathsf{P}(R_s)$ be defined as above. Then $\mathsf{Obs}(R_s)$ is the TRS over $\Sigma_d \cup \Sigma_s \cup \{:, \mathsf{head}, \mathsf{tail}\}$ consisting of*

- *the two rules*
$$\mathsf{head}(x : \sigma) \to x, \quad \mathsf{tail}(x : \sigma) \to \sigma,$$

- *for every rule in $\mathsf{P}(R_s)$ of the shape $\ell \to u : t$ the two rules*
$$\mathsf{head}(\ell) \to u, \quad \mathsf{tail}(\ell) \to t,$$

- *for every rule in $\mathsf{P}(R_s)$ of the shape $\ell \to r$ with $\mathsf{root}(r) \neq :$ the two rules*
$$\mathsf{head}(\ell) \to \mathsf{head}(r), \quad \mathsf{tail}(\ell) \to \mathsf{tail}(r).$$

Example 2. For the TRS R_s given in Example 1 we rename the symbol tail by $\mathsf{tail0}$ in order to keep the symbol tail for the fresh symbol introduced in the Obs construction. Then the TRS $\mathsf{Obs}(R_s)$ consists of the following rules:

$$
\begin{aligned}
\mathsf{head}(x : \sigma) &\to x & \mathsf{head}(\mathsf{tail0}(\sigma)) &\to \mathsf{head}(\mathsf{tail}(\sigma)) \\
\mathsf{tail}(x : \sigma) &\to \sigma & \mathsf{tail}(\mathsf{tail0}(\sigma)) &\to \mathsf{tail}(\mathsf{tail}(\sigma)) \\
\mathsf{head}(\mathsf{morse}) &\to 0 & \mathsf{head}(\mathsf{zip}(\sigma, \tau)) &\to \mathsf{head}(\sigma) \\
\mathsf{tail}(\mathsf{morse}) &\to \mathsf{zip}(\mathsf{inv}(\mathsf{morse}), \mathsf{tail}(\mathsf{morse})) & \mathsf{tail}(\mathsf{zip}(\sigma, \tau)) &\to \mathsf{zip}(\tau, \mathsf{tail}(\sigma)) \\
\mathsf{head}(\mathsf{inv}(\sigma)) &\to \mathsf{not}(\mathsf{head}(\sigma)) \\
\mathsf{tail}(\mathsf{inv}(\sigma)) &\to \mathsf{inv}(\mathsf{tail}(\sigma))
\end{aligned}
$$

Together with the rules $\mathsf{not}(0) \to 1$ and $\mathsf{not}(1) \to 0$ from R_d this TRS is terminating as can easily be proved fully automatically by AProVE [4] or TTT2 [7]. As a consequence, the result of this paper states that the specification uniquely defines a stream for every ground term of type s, in particular for morse.

4   The Main Theorem

We start this section by presenting our main theorem.

Theorem 1. *Let $(\Sigma_d, \Sigma_s, R_d, R_s)$ be a stream specification for which the TRS $\mathsf{Obs}(R_s) \cup R_d$ is terminating. Then the stream specification admits a unique model $(S, ([f])_{f \in \Sigma_s})$ satisfying $S = \{[t] \mid t \in \mathcal{T}_s\}$.*

Before proving the theorem we show by an example why it is essential to restrict to $S = \{[t] \mid t \in \mathcal{T}_s\}$ rather than choosing $S = D^\omega$. A degenerate example is obtained if there are no constants of sort s, and hence $\mathcal{T}_s = \emptyset$. More interesting is the following.

Example 3. Let the boolean stream specification consist of $R_d = \emptyset$ and R_s consisting of the following rules:

$$c \to 1 : c \qquad\qquad g(0, \sigma) \to f(\sigma)$$
$$f(x : \sigma) \to g(x, \sigma) \qquad g(1, \sigma) \to 1 : f(\sigma)$$

So f tries to remove all 0's from its argument. For streams containing infinitely many 0's this may be problematic. Note that by the symbols $c, :, 0$ and 1 only the streams with finitely many 0's can be constructed, for ground terms this problem does not arise. Indeed the TRS $\mathsf{Obs}(R_s) \cup R_d$ is terminating, and by Theorem 1 the specification admits a unique model $(S, ([f])_{f \in \Sigma_s})$ satisfying $S = \{[t] \mid t \in \mathcal{T}_s\}$. However, when extending to all streams the function $[f] : D^\omega \to D^\omega$ is not uniquely defined, even if we strengthen the requirement of $[\ell\rho] = [r\rho]$ for all rules $\ell \to r$ and all ground substitutions ρ to an open variant in which the σ's in the rules are replaced by arbitrary streams. Write ones and zeros for the streams only consisting of ones, resp. zeros. Two distinct models $[\cdot]_1$ and $[\cdot]_2$ satisfying the stream specification are defined by:

$$[c]_1 = [f]_1(s) = [g]_1(u, s) = \mathsf{ones} \text{ for all } s \in D^\omega, u \in D,$$

and $[c]_2 = [f]_2(s) = [g]_2(u, s) = \mathsf{ones}$ for $u \in D$ and streams s containing infinitely many ones, and $[f]_2(s) = 1^n : \mathsf{zeros}$, $[g]_2(u, s) = [f]_2(u : s)$ for $u \in D$ and streams s containing $n < \infty$ ones.

Now we arrive at the proof of Theorem 1. The plan of the proof is as follows.

– First we construct a function $[\cdot]_1 : \mathcal{T}_s \to D^\omega$, and choose $S_1 = \{[t]_1 \mid t \in \mathcal{T}_s\}$.
– Next we show that if $[t_i]_1 = [t'_i]_1$ for $i = 1, \ldots, m$, then

$$[f(u_1, \ldots, u_n, t_1, \ldots, t_m)]_1 = [f(u_1, \ldots, u_n, t'_1, \ldots, t'_m)]_1,$$

  by which $[f]_1$ is well-defined and we have a model $(S_1, ([f]_1)_{f \in \Sigma_s})$.
– We show this model satisfies the specification.
– We show no other model $(S, ([f])_{f \in \Sigma_s})$ satisfies the specification and $S = \{[t] \mid t \in \mathcal{T}_s\}$.

First we define $[t]_1 \in D^\omega$ for any $t \in \mathcal{T}_s$. Since elements of D^ω are functions from $\mathbf{N}$ to D, a function $[t]_1 \in D^\omega$ is defined by defining $[t]_1(n)$ for every $n \in \mathbf{N}$. Due to the assumption of the theorem the TRS $\mathsf{Obs}(R_s) \cup R_d$ is terminating. According to the definition of stream specification the TRS $R_s \cup R_d$ is orthogonal, and by the construction Obs the TRS $\mathsf{Obs}(R_s) \cup R_d$ is orthogonal too. So every ground term of sort d has a unique normal form with respect to $\mathsf{Obs}(R_s) \cup R_d$.

Assume such a normal form contains a symbol from $\Sigma_s \cup \{:\}$. Choose such a symbol with minimal position, that is, closest to the root. Since the term is

of sort d, this symbol is not the root. Hence it has a parent. Due to minimality of position, this parent is either head or tail. Due to the shape of the rules of $\mathsf{Obs}(R_s)$, a rule of $\mathsf{Obs}(R_s)$ is applicable on this parent position, contradicting the normal form assumption. So the normal form only contains symbols from Σ_d. Since it is also a normal form with respect to R_d, such a normal form is an element of D. Now for $t \in \mathcal{T}_s$ and $n \in \mathbf{N}$ we define

$$[t]_1(n) = \text{the normal form of } \mathsf{head}(\mathsf{tail}^n(t)) \text{ with respect to } \mathsf{Obs}(R_s) \cup R_d,$$

in this way defining $[t]_1 \in D^\omega$.

Lemma 1. *Let $\mathsf{Obs}(R_s) \cup R_d$ be terminating. Let $f \in \Sigma_s$ of type $d^n \times s^m \to s$. Let $u_1, \ldots, u_n \in D$ and $t_1, \ldots, t_m, t'_1, \ldots, t'_m \in \mathcal{T}_s$ satisfying $[t_i]_1 = [t'_i]_1$ for $i = 1, \ldots, m$. Then*

$$[f(u_1, \ldots, u_n, t_1, \ldots, t_m)]_1 = [f(u_1, \ldots, u_n, t'_1, \ldots, t'_m)]_1.$$

Proof. First we extend the definition of $[\cdot]_1$ to all ground terms over $\Sigma_s \cup \Sigma_d \cup \{:, \mathsf{head}, \mathsf{tail}\}$. For ground terms t of sort s we define it by $[t]_1(n) = $ the normal form of $\mathsf{head}(\mathsf{tail}^n(t))$ with respect to $\mathsf{Obs}(R_s) \cup R_d$, and for ground terms u of sort d we define $[u]_1$ to be the normal form of u with respect to $\mathsf{Obs}(R_s) \cup R_d$. We prove the following claim.

> **Claim 1:** Let $[t]_1 = [t']_1$ for $t, t' \in \mathcal{T}_s$. Let T be a ground term over $\Sigma_s \cup \Sigma_d \cup \{:, \mathsf{head}, \mathsf{tail}\}$ of sort s containing t as a subterm. Let T' be obtained from T by replacing zero or more occurrences of the subterm t by t'. Then
> $$[\mathsf{head}(T)]_1 = [\mathsf{head}(T')]_1.$$

Let $>$ be the well-founded order on ground terms being the strict part of $\geq$ defined by

$$v \geq v' \iff v' \text{ is a subterm } v'' \text{ such that } v \to^*_{\mathsf{Obs}(R_s) \cup R_d} v''.$$

We prove the claim for every such term $\mathsf{head}(T)$ by noetherian induction on $>$.

Claim 1 is trivial if $t = T$, so we may assume that $T = f(u_1, \ldots, u_n, t_1, \ldots, t_m)$ such that t occurs in $u_1, \ldots, u_n, t_1, \ldots, t_m$, and either $f \in \Sigma_s \cup \{:, \mathsf{tail}\}$, and $T' = f(u'_1, \ldots, u'_n, t'_1, \ldots, t'_m)$. For every subterm of u_i of the shape $\mathsf{head}(\cdots)$ we may apply the induction hypothesis, yielding $[u_i]_1 = [u'_i]_1 = d_i$ for all i, defining $d_i \in D$.

In case the root of T is not tail we rewrite

$$\mathsf{head}(T) \to^*_{\mathsf{Obs}(R_s) \cup R_d} \mathsf{head}(f(d_1, \ldots, d_n, t_1, \ldots, t_m))$$

by the rule $\mathsf{head}(f(\cdots)) \to \cdots$ in $\mathsf{Obs}(R_s)$, yielding a term U of sort d. The only way such a term can contain t as a subterm is by $U = C[\mathsf{head}(V_1), \ldots, \mathsf{head}(V_k)]$ where t is a subterm of some of the V_i and C is composed from Σ_d. By the induction hypothesis we obtain $[\mathsf{head}(V_i)]_1 = [\mathsf{head}(V'_i)]_1$ for V'_i obtained from V_i by

replacing zero ore more occurrences of t by t'. Hence $[\mathsf{head}(T)]_1 = C[\mathsf{head}(V_1), \ldots, \mathsf{head}(V_k)]]_1 = C[\mathsf{head}(V_1'), \ldots, \mathsf{head}(V_k')]]_1 = [\mathsf{head}(T')]_1$.

In case the root of T is tail then write $T = \mathsf{tail}^i(f(\cdots)) \rightarrow^*_{\mathsf{Obs}(R_s) \cup R_d}$ $\mathsf{tail}^i(f(d_1, \ldots, d_n, t_1, \ldots, t_m)$ for $f \in \Sigma_s \cup \{:\}$. This can be rewritten by the rule $\mathsf{tail}(f(\cdots)) \rightarrow \cdots$ in $\mathsf{Obs}(R_s)$, yielding V. On the same position using the same rule we can rewrite $T' \rightarrow_{\mathsf{Obs}(R_s)} V'$ for V' obtained from V by replacing one ore more occurrences of t by t'. Applying the induction hypothesis gives $[\mathsf{head}(V)]_1 = [\mathsf{head}(V')]_1$ yielding

$$[\mathsf{head}(T)]_1 = [\mathsf{head}(V)]_1 = [\mathsf{head}(V')]_1 = [\mathsf{head}(T')]_1,$$

concluding the proof of Claim 1.

Claim 2: Let $[t]_1 = [t']_1$ for $t, t' \in \mathcal{T}_s$. Let T be a ground term over $\Sigma_s \cup \Sigma_d \cup \{:, \mathsf{head}, \mathsf{tail}\}$ of sort s containing t as a subterm. Let T' be obtained from T by replacing one or more occurrences of the subterm t by t'. Then $[T]_1 = [T']_1$.

Claim 2 easily follows from Claim 1 and the observation

$$[T]_1 = [T']_1 \iff \forall i \in \mathbf{N} : [\mathsf{head}(\mathsf{tail}^i(T))]_1 = [\mathsf{head}(\mathsf{tail}^i(T'))]_1.$$

Now the lemma follows by applying Claim 2 and replacing t_i by t_i' successively for $i = 1, \ldots, m$. □

Define $S_1 = \{[t]_1 \mid t \in \mathcal{T}_s\}$. For any $f \in \Sigma_s$ of type $d^n \times s^m \rightarrow s$ for $u_1, \ldots, u_n \in D$ and $t_1, \ldots, t_m, t_1', \ldots, t_m' \in \mathcal{T}_s$ we now define $[f]_1 : D^n \times S^m \rightarrow S$ by

$$[f]_1(u_1, \ldots, u_n, [t_1], \ldots, [t_m]) = [f(u_1, \ldots, u_n, t_1, \ldots, t_m)]_1;$$

Lemma 1 implies that this is well-defined: the result is independent of the choice of the representants in $[t_i]_1$. So $(S_1, ([f]_1)_{f \in \Sigma_s})$ is a model.

Next we will prove that it satisfies the specification, and essentially is the only one doing so.

Lemma 2. *Let $\ell \rightarrow r \in R_s$ and let ρ be a substitution. Then*

- *there is a term t such that $\mathsf{head}(\ell\rho) \rightarrow^*_{\mathsf{Obs}(R_s)} t$ and $\mathsf{head}(r\rho) \rightarrow^*_{\mathsf{Obs}(R_s)} t$, and*
- *there is a term t such that $\mathsf{tail}(\ell\rho) \rightarrow^*_{\mathsf{Obs}(R_s)} t$ and $\mathsf{tail}(r\rho) \rightarrow^*_{\mathsf{Obs}(R_s)} t$.*

Proof. Let f be the root of ℓ. Define ρ' by $\sigma\rho' = x\rho : \sigma\rho$ for every argument of the shape $x : \sigma$ of f in ℓ, and ρ' coincides with ρ on all other variables. Then $\mathsf{head}(\ell\rho) = \ell'\rho'$ for some rule in $\ell' \rightarrow r'$ in $\mathsf{Obs}(R_s)$. Now a common reduct t of $r'\rho'$ and $\mathsf{head}(r\rho)$ is obtained by applying the rule $\mathsf{head}(x : \sigma) \rightarrow x$ zero or more times. This yields $\mathsf{head}(\ell\rho) = \ell'\rho' \rightarrow_{\mathsf{Obs}(R_s)} r'\rho' \rightarrow^*_{\mathsf{Obs}(R_s)} t$ and $\mathsf{head}(r\rho) \rightarrow^*_{\mathsf{Obs}(R_s)} t$. The argument for $\mathsf{tail}(\ell\rho)$ and $\mathsf{tail}(r\rho)$ is similar. □

Lemma 3. *The model* $(S_1, ([f]_1)_{f \in \Sigma_s})$ *satisfies the specification* $(\Sigma_d, \Sigma_s, R_d, R_s)$.

Proof. We have to prove that $[\ell\rho]_1(i) = [r\rho]_1(i)$ for every rule $\ell \to r$ in R_s, every ground substitution ρ and every $i \in \mathbf{N}$. By definition $[\ell\rho]_1(i)$ is the unique normal form with respect to $\mathsf{Obs}(R_s) \cup R_d$ of $\mathsf{head}(\mathsf{tail}^i(\ell\rho))$, and $[r\rho]_1(i)$ is the similar normal form of $\mathsf{head}(\mathsf{tail}^i(r\rho))$. Now the lemma follows from Lemma 2. $\square$

For concluding the proof of Theorem 1 we have to prove that $(S_1, ([f]_1)_{f \in \Sigma_s})$ is the only model satisfying the specification $(\Sigma_d, \Sigma_s, R_d, R_s)$ and $S = \{[t] \mid t \in \mathcal{T}_s\}$. This follows from the following lemma.

Lemma 4. *Let* $(S, ([f])_{f \in \Sigma_s})$ *be any model satisfying* $(\Sigma_d, \Sigma_s, R_d, R_s)$, *and* $t \in \mathcal{T}_s$. *Then* $[t] = [t]_1$.

Proof. By definition in the model for $u \in D$ and $s \in S$ we have

$$([:](u, s))(0) = u, \quad ([:](u, s))(i) = s(i - 1) \text{ for } i > 0.$$

In the original stream specification the symbols $\mathsf{head}, \mathsf{tail}$ do not occur, for these fresh symbols we now define functions $[\mathsf{head}]$ and $[\mathsf{tail}]$ on streams s by

$$[\mathsf{head}](s) = s(0), \quad ([\mathsf{tail}](s))(i) = s(i + 1) \text{ for } i \geq 0.$$

If $S \neq D^\omega$ then it is not clear whether $[\mathsf{tail}](s) \in S$ for every $s \in S$. Therefore we extend S to D^ω and define $[f](\cdots)$ to be any arbitrary value if at least one argument is in $D^\omega \setminus S$; note that for the model satisfying the specification we only required $[\ell\rho] = [r\rho]$ for ground substitutions to $\mathcal{T}_s$ by which these junk values do not play a role.

Due to the definitions of $[:], [\mathsf{head}]$ and $[\mathsf{tail}]$ this extended model satisfies the equations

$$\mathcal{E} = \begin{cases} \mathsf{head}(x : \sigma) = x \\ \mathsf{tail}(x : \sigma) = \sigma \\ \qquad \sigma = \mathsf{head}(\sigma) : \mathsf{tail}(\sigma) \end{cases}$$

that is, for ρ mapping x to any term of sort d and σ to any term of sort s we have $[\ell\rho] = [r\rho]$ for every $\ell \to r \in \mathcal{E}$. From the definition of $\mathsf{Obs}(R_s)$ it is easily checked that any innermost step $t \to_{\mathsf{Obs}(R_s)} t'$ on a ground term t is either an application of one of the first two rules of $\mathcal{E}$, or it is of the shape

$$t \to_{\mathcal{E}}^* \cdot \to_{R_s} \cdot \to_{\mathcal{E}}^* t'$$

where due to the innermost requirement the redex of the $\to_{R_s}$ step does not contain the symbols head or tail so is in $\mathcal{T}_s$. Since the model is assumed to satisfy the specification $(\Sigma_d, \Sigma_s, R_d, R_s)$, we conclude that $[t] = [t']$ for every innermost ground step $t \to_{\mathsf{Obs}(R_s)} t'$.

For the lemma we have to prove that $[t](i) = [t]_1(i)$ for every $i \in \mathbf{N}$. By definition $[t]_1(i)$ is the normal form with respect to $\mathsf{Obs}(R_s) \cup R_d$ of $\mathsf{head}(\mathsf{tail}^i(t))$.

Now consider an innermost $\mathsf{Obs}(R_s) \cup R_d$-reduction of $\mathsf{head}(\mathsf{tail}^i(t))$ to $[t]_1(i)$. By the above observation and the definitions of $[\mathsf{head}]$ and $[\mathsf{tail}]$ we conclude that

$$[t](i) = [\mathsf{head}(\mathsf{tail}^i(t))] = [[t]_1(i)] = [t]_1(i),$$

the last step since $[t]_1(i) \in D$. This concludes the proof, both of the lemma and Theorem 1.   □

We conclude this section by an example of a well-defined stream specification that is not productive.

Example 4. Choose $\Sigma_s = \{c, f, g\}$, $\Sigma_d = \{0, 1\}$, $R_d = \emptyset$, and R_s consists of the following rules:

$$c = 1 : c$$
$$f(x : \sigma) = g(f(\sigma))$$
$$g(x : \sigma) = c.$$

Then this is a valid stream specification for which $\mathsf{Obs}(R_s)$ is terminating, as can be shown by AProVE [4] or TTT2 [7]. Hence by Theorem 1 there is a unique model. So the ground term $f(c)$ has a unique interpretation: the stream only consisting of 1's. However, $f(c)$ is not productive.

So the TRS R_s is not suitable to compute the interpretation of $f(c)$. Instead one can use outermost reduction with respect to $\mathsf{P}(R_s)$, where $\mathsf{P}(R_s)$ is the TRS introduced in the definition of $\mathsf{Obs}(R_s)$.

5   Data Independent Stream Functions

The reason that in Theorem 1 we have to restrict to models satisfying $S = \{[t] \mid t \in \mathcal{T}_s\}$, as we saw in Example 3, is in the fact that computations may be guarded by data elements in left hand sides of rules. Next we show that we also get well-definedness for stream functions defined on all streams in case the left hand sides of the rule do not contain data elements.

Theorem 2. *Let* $(\Sigma_d, \Sigma_s, R_d, R_s)$ *be a stream specification for which the TRS* $\mathsf{Obs}(R_s) \cup R_d$ *is terminating and the only subterms of left hand sides of* R_s *of sort d are variables. Then the stream specification admits a unique model* $(S, ([f])_{f \in \Sigma_s})$ *satisfying* $S = D^\omega$.

Proof. (sketch) We have to prove that for any $f \in \Sigma_s$ of type $d^n \times s^m \to s$ the function $[f] : D^n \times (D^\omega)^m \to D^\omega$ is uniquely defined. For doing so we introduce m fresh constants $c_1, \ldots, c_m$ of sort s. Let $k \in \mathbf{N}$ and $u_1, \ldots, u_n \in D$. Due to termination and orthogonality of $\mathsf{Obs}(R_s) \cup R_d$, the term

$$\mathsf{head}(\mathsf{tail}^k(f(u_1, \ldots, u_n, c_1, \ldots, c_m)))$$

has a unique normal for with respect to $\mathsf{Obs}(R_s) \cup R_d$. Since it is of sort d, due to the shape of the rules it is a ground term of sort d over $\Sigma_d \cup \{\mathsf{head}, \mathsf{tail}, c_1, \ldots, c_m\}$, that is, a ground term T composed from Σ_d and terms of the shape $\mathsf{head}(\mathsf{tail}^i(c_j))$

for $i \in \mathbf{N}$ and $j \in \{1,\ldots,m\}$. For this observation it is essential that left hand sides do not contain non-variable terms of sort d: terms of the shape $f(\mathsf{head}(\cdots),\ldots)$ should be rewritten.

Let N be the greatest number i for which T has a subterm of the shape $\mathsf{head}(\mathsf{tail}^i(c_j))$. Let $s_1,\ldots,s_m \in D^\omega$. Define $t_j = s_j(0) : s_j(1) : \cdots : s_j(N) : \sigma$. Since $\mathsf{head}(\mathsf{tail}^k(f(u_1,\ldots,u_n,c_1,\ldots,c_m)))$ rewrites to T, the term $\mathsf{head}(\mathsf{tail}^k(f(u_1,\ldots,u_n,t_1,\ldots,t_m)))$ rewrites to T' obtained from T by replacing every subterm of the shape $\mathsf{head}(\mathsf{tail}^i(c_j))$ by $\mathsf{head}(\mathsf{tail}^i(t_j))$. Observe that $\mathsf{head}(\mathsf{tail}^i(t_j))$ rewrites to $s_j(i) \in D$. So $([f](u_1,\ldots,u_n,s_1,\ldots,s_m))(k)$ has to be the R_d-normal form of the ground term over Σ_d obtained from T by replacing every subterm of the shape $\mathsf{head}(\mathsf{tail}^i(c_j))$ by $s_j(i) \in D$. Since this fixes $([f](u_1,\ldots,u_n,s_1,\ldots,s_m))(k)$ for every k, this uniquely defines $[f]$.   □

Example 5. It is easy to see that for the standard stream functions zip, even and odd defined by

$$\mathsf{even}(x : \sigma) = x : \mathsf{odd}(\sigma), \quad \mathsf{odd}(x : \sigma) = \mathsf{even}(\sigma), \quad \mathsf{zip}(x : \sigma, \tau) = x : \mathsf{zip}(\tau, \sigma),$$

there exists $f : D^\omega \to D^\omega$ for every data set D satisfying

$$f(x : \sigma) \;=\; x : \mathsf{zip}(f(\mathsf{even}(\sigma)), f(\mathsf{odd}(\sigma))),$$

namely the identity. By Theorem 2 we can conclude it is the only one, since for $R_d = \emptyset$ and R_s consisting of the above four rules, the resulting TRS $\mathsf{Obs}(R_s)$ is terminating as can be proved by AProVE [4] or TTT2 [7]. Both [3] and [11] fail to prove that the identity is the only stream function satisfying the equation for f. By essentially choosing $\mathsf{Obs}(R_s)$ as the input and adding information about special contexts, the tool Circ [8] is able to prove that f is the identity.

6   Fixpoints

Several streams are defined as fixpoints of stream functions, like the Fibonacci stream as given in the introduction. In our format it can be presented as the stream specification R_s consisting of the rules

$$\mathsf{Fib} = f(\mathsf{Fib}) \qquad\qquad g(0, \sigma) = 0 : 1 : f(\sigma)$$
$$f(x : \sigma) = g(x, \sigma) \qquad\qquad g(1, \sigma) = 0 : f(\sigma).$$

The TRS $\mathsf{Obs}(R_s)$ is not terminating since it allows the reduction

$$\mathsf{head}(\mathsf{Fib}) \to \mathsf{head}(f(\mathsf{Fib})) \to \mathsf{head}(g(\mathsf{head}(\mathsf{Fib}), \mathsf{tail}(\mathsf{Fib}))).$$

However, now we will polish R_s to R'_s such that $\mathsf{Obs}(R'_s)$ is terminating, by which well-definedness of both R'_s and R_s can be concluded. This shows incompleteness of Theorem 1: the stream specification R_s admits a unique model but $\mathsf{Obs}(R_s)$ is not terminating.

Assume some model satisfies R_s; for simplicity we identify ground terms with their interpretations in the model. Then $\mathsf{Fib} = f(\mathsf{Fib}) = g(\cdots) = 0 : c$ for some stream c. Using this equality $\mathsf{Fib} = 0 : c$ we obtain

$$0 : c = \mathsf{Fib} = f(\mathsf{Fib}) = f(0 : c) = 0 : 1 : f(c),$$

so $c = 1 : f(c)$. So the model also satisfies R'_s which is obtained from R_s by replacing the first rule $\mathsf{Fib} = f(\mathsf{Fib})$ by the two rules $\mathsf{Fib} = 0 : c$ and $c = 1 : f(c)$. However, R'_s again satisfies our format and $\mathsf{Obs}(R'_s)$ is terminating as can be proved by APROVE [4] or TTT2 [7]. So by Theorem 1 R'_s admits a unique model, which is by construction the only model for R_s too.

This technique of modifying the stream specification is generally applicable. If our technique fails for proving well-definedness of a stream specification, we can analyze the specified streams by applying the rules and deriving new equalities from which a modified stream specification can be composed. If our technique succeeds in proving well-definedness of the modified specification, conclusions can be drawn about the original one.

In general, stream functions may have zero, one or several fixpoints. For instance, the boolean stream function f defined by

$$f(0 : \sigma) = 0 : 1 : f(\sigma), \qquad f(1 : \sigma) = 1 : 0 : f(\sigma),$$

has two fixpoints: the Thue Morse stream morse from Example 1 and its inverse. Proving that there are exactly two can be done as follows. Assume m is a fixpoint starting with 0, so $m = 0 : c$. Then $0 : c = m = f(m) = f(0 : c) = 0 : 1 : f(c)$, so $c = 1 : f(c)$. By adding the rules $m = 0 : c$ and $c = 1 : f(c)$ we have a stream specification R_s for which termination of $\mathsf{Obs}(R_s)$ can be proved. So there is exactly one fixpoint of f starting with 0, and by symmetry there is exactly one fixpoint of f starting with 1.

7   Conclusions

We presented a technique by which well-definedness of stream specifications like

$$\begin{aligned} f(0 : \sigma) &= 1 : f(\sigma) \\ f(1 : \sigma) &= 0 : f(f(\sigma)) \\ c &= 1 : c \end{aligned}$$

can be proved fully automatically, where a tool like Circ [8] fails, and the productivity tool [3] fails to prove productivity of $f(c)$. The main idea is to prove well-definedness by proving termination of a transformed system $\mathsf{Obs}(R_s)$, in this way exploiting the power of present termination provers.

We observed that productivity of the stream specification can not be concluded from termination of $\mathsf{Obs}(R_s)$; we leave as a challenge to find syntactic criteria on the stream specification by which this can be concluded.

References

1. Allouche, J.-P., Shallit, J.: Automatic Sequences: Theory, Applications, Generalizations. Cambridge University Press, Cambridge (2003)
2. Arts, T., Giesl, J.: Termination of term rewriting using dependency pairs. Theoretical Computer Science 236, 133–178 (2000)
3. Endrullis, J., Grabmayer, C., Hendriks, D.: Data-oblivious stream productivity. In: Cervesato, I., Veith, H., Voronkov, A. (eds.) LPAR 2008. LNCS, vol. 5330, pp. 79–96. Springer, Heidelberg (2008), webinterface tool: `http://fspc282.few.vu.nl/productivity/`
4. Giesl, J., et al.: Automated program verification environment (AProVE), `http://aprove.informatik.rwth-aachen.de/`
5. Giesl, J., Thiemann, R., Schneider-Kamp, P.: The dependency pair framework: Combining techniques for automated termination proofs. In: Baader, F., Voronkov, A. (eds.) LPAR 2004. LNCS, vol. 3452, pp. 301–331. Springer, Heidelberg (2005)
6. Goguen, J., Lin, K., Rosu, G.: Circular coinductive rewriting. In: Proceedings, 15th International Conference on Automated Software Engineering (ASE 2000), Grenoble, France, September 11-15, 2000, Institute of Electrical and Electronics Engineers Computer Society (2000), webinterface tool CIRC: `http://fsl.cs.uiuc.edu/index.php/Special:CircOnline`
7. Korp, M., Sternagel, C., Zankl, H., Middeldorp, A.: Tyrolean termination tool 2. In: Treinen, R. (ed.) RTA 2009. LNCS, vol. 5595, pp. 295–304. Springer, Heidelberg (2009), Tool available at: `http://colo6-c703.uibk.ac.at/ttt2/`
8. Lucanu, D., Rosu, G.: CIRC: A circular coinductive prover. In: Mossakowski, T., Montanari, U., Haveraaen, M. (eds.) CALCO 2007. LNCS, vol. 4624, pp. 372–378. Springer, Heidelberg (2007)
9. Marche, C., Zantema, H.: The termination competition. In: Baader, F. (ed.) RTA 2007. LNCS, vol. 4533, pp. 303–313. Springer, Heidelberg (2007)
10. Roşu, G.: Equality of streams is a Π_2^0-complete problem. In: Proceedings of the 11th ACM SIGPLAN International Conference on Functional Programming (ICFP 2006). ACM Press, New York (2006)
11. Rutten, J.J.M.M.: A coinductive calculus of streams. Mathematical Structures in Computer Science 15, 93–147 (2005)
12. Simonsen, J.G.: The Π_2^0-completeness of most of the properties of rewriting systems you care about (and productivity). In: Treinen, R. (ed.) RTA 2009. LNCS, vol. 5595, pp. 335–349. Springer, Heidelberg (2009)