A Tool proving Well-definedness of Streams using Termination Tools

Hans Zantema

Department of Computer Science, TU Eindhoven, and
Computing Science Department, Radboud University Nijmegen
The Netherlands
email: h.zantema@tue.nl

presented by Eugen Goriac

CALCO, 2009

Stream Specifications: Example

```
 0 0...      = P
0 1...       = alt
0010...      = P (zipping)
```

Stream Specifications: Example

` 0 0...`	$= P$
`0 1...`	$=$ alt
`0010...`	$= P$ (zipping)
` 0 0 1 0...`	$= P$
`0 1 0 1...`	$=$ alt
`00100110...`	$= P$ (zipping)

Stream Specifications: Example

```
 0 0...                     = P
0 1...                      = alt
0010...                     = P (zipping)
 0 0 1 0...                 = P
0 1 0 1...                  = alt
00100110...                 = P (zipping)
 0 0 1 0 0 1 1 0...         = P
0 1 0 1 0 1 0 1...          = alt
0010011000110110...         = P (zipping)
```

Stream Specifications: Example

`  0   0...`	$= P$
`0   1...`	$=$ alt
`0010...`	$= P$ (zipping)
`  0   0   1   0...`	$= P$
`0   1   0   1...`	$=$ alt
`00100110...`	$= P$ (zipping)
`  0   0   1   0   0   1   1   0...`	$= P$
`0   1   0   1   0   1   0   1...`	$=$ alt
`0010011000110110...`	$= P$ (zipping)
`  0   0   1   0   0   1   1   0   0   0   1   1   0   1   1   0...`	$= P$
`0   1   0   1   0   1   0   1   0   1   0   1   0   1   0   1...`	$=$ alt
`00100110001101100010011100110110...`	$= P$ (zipping)

Stream Specifications: Example

`  0   0...`	$= P$
`0   1...`	$=$ alt
`0010...`	$= P$ (zipping)
`  0   0   1   0...`	$= P$
`0   1   0   1...`	$=$ alt
`00100110...`	$= P$ (zipping)
`  0   0   1   0   0   1   1   0...`	$= P$
`0   1   0   1   0   1   0   1...`	$=$ alt
`0010011000110110...`	$= P$ (zipping)
`  0   0   1   0   0   1   1   0   0   0   1   1   0   1   1   0...`	$= P$
`0   1   0   1   0   1   0   1   0   1   0   1   0   1   0   1...`	$=$ alt
`00100110001101100010011100110110...`	$= P$ (zipping)

$$P = \mathrm{zip}(\mathrm{alt}, P)$$

$$\mathrm{alt} = 0 : 1 : \mathrm{alt}$$

$$\mathrm{zip}(x : \sigma, \tau) = x : \mathrm{zip}(\tau, \sigma)$$

The Problem

Given:

$$P = \mathrm{zip}(\mathrm{alt}, P)$$
$$\mathrm{alt} = 0 : 1 : \mathrm{alt}$$
$$\mathrm{zip}(x : \sigma, \tau) = x : \mathrm{zip}(\tau, \sigma)$$

Question:
Does this set of equations uniquely define P?

The Problem

Given:

$$P = \mathrm{zip}(\mathrm{alt}, P)$$
$$\mathrm{alt} = 0 : 1 : \mathrm{alt}$$
$$\mathrm{zip}(x : \sigma, \tau) = x : \mathrm{zip}(\tau, \sigma)$$

Question:
Does this set of equations uniquely define P?

This is not obvious: we give sets of similar equations not having a unique stream solution

Counter-examples

The set of equations

$$M = f(M)$$

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

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

Counter-examples

The set of equations

$$M = f(M)$$

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

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

has exactly two solutions for M: the Thue-Morse stream and its inverse: $morse = 0 : zip(\overline{morse}, tail(morse))$

Counter-examples

The set of equations

$$M = f(M)$$

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

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

has exactly two solutions for M: the Thue-Morse stream and its inverse: $morse = 0 : zip(\overline{morse}, tail(morse))$

The set of equations

$$M = f(M)$$

$$f(x : \sigma) = 0 : \sigma$$

Counter-examples

The set of equations

$$M = f(M)$$

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

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

has exactly two solutions for M: the Thue-Morse stream and its inverse: $morse = 0 : zip(\overline{morse}, tail(morse))$

The set of equations

$$M = f(M)$$

$$f(x : \sigma) = 0 : \sigma$$

has infinitely many solutions for M: every stream starting with 0

The Solution 1/2

When is a stream uniquely defined?

The Solution 1/2

When is a stream uniquely defined?

If the *n*-th element of the stream is uniquely determined, for every *n*

The Solution 1/2

When is a stream uniquely defined?

If the *n*-th element of the stream is uniquely determined, for every *n*

So we need a deterministic computation to compute the *n*-th element of the stream

The Solution 1/2

When is a stream uniquely defined?

If the *n*-th element of the stream is uniquely determined, for every *n*

So we need a deterministic computation to compute the *n*-th element of the stream

A possible solution: terminating and confluent/orthogonal term rewriting system (TRS)

The Solution 1/2

When is a stream uniquely defined?

If the *n*-th element of the stream is uniquely determined, for every *n*

So we need a deterministic computation to compute the *n*-th element of the stream

A possible solution: terminating and confluent/orthogonal term rewriting system (TRS)

Interpreted as TRSs our sets of equations are orthogonal, but not terminating:
computing the whole stream will never be terminating

The Solution 2/2

We need to compute / observe the n-th element, for every n

The Solution 2/2

We need to compute / observe the n-th element, for every n

Introduce

- head to give the first element of a stream
- tail to remove the first element of a stream

The Solution 2/2

We need to compute / observe the *n*-th element, for every *n*

Introduce

- head to give the first element of a stream
- tail to remove the first element of a stream

Then the *n*-the element of a stream σ can be expressed as

$$\text{head}(\text{tail}^{n-1}(\sigma))$$

The Solution 2/2

We need to compute / observe the n-th element, for every n

Introduce

- head to give the first element of a stream
- tail to remove the first element of a stream

Then the n-the element of a stream σ can be expressed as

$$\mathrm{head}(\mathrm{tail}^{n-1}(\sigma))$$

We will modify the TRS R representing our equations to an observational variant $\mathrm{Obs}(R)$ by which the value of $\mathrm{head}(\mathrm{tail}^{n-1}(t))$ can be computed for every n and every ground term t

The Solution 2/2

We need to compute / observe the n-th element, for every n

Introduce

- head to give the first element of a stream
- tail to remove the first element of a stream

Then the n-the element of a stream σ can be expressed as

$$\text{head}(\text{tail}^{n-1}(\sigma))$$

We will modify the TRS R representing our equations to an observational variant Obs(R) by which the value of $\text{head}(\text{tail}^{n-1}(t))$ can be computed for every n and every ground term t

If Obs(R) is terminating and orthogonal, then all streams represented by ground terms are uniquely defined

The transformation Obs

Two steps:

The transformation Obs

Two steps:

- First we replace every $x : \sigma$ in left hand sides of rules by a fresh variable σ, and replace
  - x by head(σ), and
  - original σ by tail(σ)

The transformation Obs

Two steps:

- First we replace every $x : \sigma$ in left hand sides of rules by a fresh variable σ, and replace
  - x by head(σ), and
  - original σ by tail(σ)

So $\mathrm{zip}(x : \sigma, \tau) \to x : \mathrm{zip}(\tau, \sigma)$ is replaced by

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

The transformation Obs

Two steps:

- First we replace every $x : \sigma$ in left hand sides of rules by a fresh variable σ, and replace
  - x by head(σ), and
  - original σ by tail(σ)

  So zip($x : \sigma, \tau$) $\to$ x : zip(τ, σ) is replaced by

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

- Next we replace rules of the shape $\ell \to u : t$ by the two rules head(ℓ) $\to u$, tail(ℓ) $\to t$,

The transformation Obs

Two steps:

- First we replace every $x : \sigma$ in left hand sides of rules by a fresh variable σ, and replace
  - x by $\mathsf{head}(\sigma)$, and
  - original σ by $\mathsf{tail}(\sigma)$

  So $\mathsf{zip}(x : \sigma, \tau) \to x : \mathsf{zip}(\tau, \sigma)$ is replaced by

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

- Next we replace rules of the shape $\ell \to u : t$ by the two rules $\mathsf{head}(\ell) \to u, \quad \mathsf{tail}(\ell) \to t$,

  other rules of the shape $\ell \to r$ by the two rules

  $$\mathsf{head}(\ell) \to \mathsf{head}(r), \quad \mathsf{tail}(\ell) \to \mathsf{tail}(r)$$

The transformation Obs

Two steps:

- First we replace every $x : \sigma$ in left hand sides of rules by a fresh variable σ, and replace
  - x by $\mathrm{head}(\sigma)$, and
  - original σ by $\mathrm{tail}(\sigma)$

  So $\mathrm{zip}(x : \sigma, \tau) \to x : \mathrm{zip}(\tau, \sigma)$ is replaced by

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

- Next we replace rules of the shape $\ell \to u : t$ by the two rules $\mathrm{head}(\ell) \to u, \quad \mathrm{tail}(\ell) \to t$,

  other rules of the shape $\ell \to r$ by the two rules

  $$\mathrm{head}(\ell) \to \mathrm{head}(r), \quad \mathrm{tail}(\ell) \to \mathrm{tail}(r)$$

  and add the two rules

  $$\mathrm{head}(x : \sigma) \to x, \quad \mathrm{tail}(x : \sigma) \to \sigma$$

Example

The TRS R:

$$\begin{array}{rcl} P & \to & \mathrm{zip}(\mathrm{alt}, P) \\ \mathrm{alt} & \to & 0 : 1 : \mathrm{alt} \\ \mathrm{zip}(x : \sigma, \tau) & \to & x : \mathrm{zip}(\tau, \sigma) \end{array}$$

Example

The TRS R:

$$\begin{aligned} P &\to \mathrm{zip}(\mathrm{alt}, P) \\ \mathrm{alt} &\to 0 : 1 : \mathrm{alt} \\ \mathrm{zip}(x : \sigma, \tau) &\to x : \mathrm{zip}(\tau, \sigma) \end{aligned}$$

transforms to Obs(R):

$$\begin{aligned} \mathrm{head}(x : \sigma) &\to x \\ \mathrm{tail}(x : \sigma) &\to \sigma \\ \mathrm{head}(P) &\to \mathrm{head}(\mathrm{zip}(\mathrm{alt}, P)) \\ \mathrm{tail}(P) &\to \mathrm{tail}(\mathrm{zip}(\mathrm{alt}, P)) \\ \mathrm{head}(\mathrm{zip}(\sigma, \tau)) &\to \mathrm{head}(\sigma) \\ \mathrm{tail}(\mathrm{zip}(\sigma, \tau)) &\to \mathrm{zip}(\tau, \mathrm{tail}(\sigma)) \\ \mathrm{head}(\mathrm{alt}) &\to 0 \\ \mathrm{tail}(\mathrm{alt}) &\to 1 : \mathrm{alt} \end{aligned}$$

The main theorem

The main theorem

A *stream specification* is a TRS such that

- every left hand side is of the shape

$$f(\cdots) \to \cdots$$

  where every stream type argument of f is either a variable σ or of the shape $x : \sigma$, and

The main theorem

A *stream specification* is a TRS such that

- every left hand side is of the shape

  $$f(\cdots) \rightarrow \cdots$$

  where every stream type argument of f is either a variable σ or of the shape $x : \sigma$, and
- every stream type ground term matches with exactly one left hand side

The main theorem

A *stream specification* is a TRS such that

- every left hand side is of the shape

  $$f(\cdots) \to \cdots$$

  where every stream type argument of f is either a variable σ or of the shape $x : \sigma$, and

- every stream type ground term matches with exactly one left hand side

Theorem

Let R be a stream specification for which $\mathrm{Obs}(R)$ is terminating. Then for every ground term the corresponding stream is uniquely defined.

Details in [H. Zantema, Well-definedness of Streams by Termination, RTA2009, LNCS 5595].

The Tool

- implements the transformation Obs
- different options for the output:
  - AProVE: the termination is proved using the AProVE tool
  - CIRC:
    – Obs(R) described as a CIRC theory
    – well-definedness can be proved using CIRC by showing that two copies of the definitions are behavioral equivalent
  - both AProVE and CIRC have Web interfaces

The tool can be downloaded from

www.win.tue.nl/~hzantema/str.zip

Demo