Lecture two:

A Coinductive Calculus of Streams

Jan Rutten

CWI Amsterdam & Radboud University Nijmegen

IPM, Tehran - 13 January 2016

Overview of this talk

1. Stream differential equations (SDEs)
2. Solving systems of SDEs
3. Formats for SDEs
4. Streams and coinduction
5. Discussion

1. Stream differential equations

Streams are the canonical example of a (final) coalgebra.

Stream differential equations:

- General framework for defining streams.
- Hand in hand with **coinduction** as main proof method.
- Ultimately leading to efficient algorithmics and automated proofs.

1. Stream differential equations

Streams are the canonical example of a (final) coalgebra.

Stream differential equations:

- General framework for defining streams.
- Hand in hand with **coinduction** as main proof method.
- Ultimately leading to efficient algorithmics and automated proofs.

Stream Differential Equations (SDEs)

We shall explain how the following diagram

$$\begin{tikzcd} X \arrow[r, dashed, "\exists!\, f"] \arrow[d, "\langle \mathrm{out}, \mathrm{tr} \rangle"'] & \mathbb{N}^\omega \arrow[d, "\langle \mathrm{head}, \mathrm{tail} \rangle"] \\ \mathbb{N} \times X \arrow[r, dashed] & \mathbb{N} \times \mathbb{N}^\omega \end{tikzcd}$$

represents a system of stream differential equations

and its solution.

A stream system/coalgebra

$$\begin{tikzcd} X \arrow[d, "\langle \mathrm{out}, \mathrm{tr} \rangle"'] \\ \mathbb{N} \times X \end{tikzcd}$$

For $x \in X$, one often writes

$$(\,\mathrm{out}(x) = n \quad \text{and} \quad \mathrm{tr}(x) = y\,) \qquad \equiv \qquad x \xrightarrow{n} y$$

(dynamical/transition system)

Stream Differential Equations

$$
\begin{tikzcd}
X \arrow[d, "\langle \mathrm{out}, \mathrm{tr} \rangle"'] \\
\mathbb{N} \times X
\end{tikzcd}
$$

Another way of writing:

$$(\mathrm{out}(x) = n \quad \text{and} \quad \mathrm{tr}(x) = y) \quad \equiv \quad (x(0) = n \quad \text{and} \quad x' = y)$$

initial value and *derivative*!

Stream Differential Equations

So we view any stream coalgebra

$$\begin{tikzcd} X \arrow[d, "\langle \mathrm{out}, \mathrm{tr} \rangle"'] \\ \mathbb{N} \times X \end{tikzcd}$$

as a **system of stream differential equations** (SDEs):

$$\{\, x(0) = \mathrm{out}(x) \quad \text{and} \quad x' = \mathrm{tr}(x) \,\}_{x \in X}$$

We think of X as the set of **variables.**

Stream Differential Equations

So we view any stream coalgebra

$$\begin{tikzcd} X \arrow[d, "\langle \mathrm{out}, \mathrm{tr} \rangle"'] \\ \mathbb{N} \times X \end{tikzcd}$$

as a **system of stream differential equations** (SDEs):

$$\{\, x(0) = \mathrm{out}(x) \quad \text{and} \quad x' = \mathrm{tr}(x) \,\}_{x \in X}$$

We think of X as the set of **variables.**

Streams

$$\begin{tikzcd} \mathbb{N}^\omega \arrow[d, "\langle \mathsf{head}, \mathsf{tail} \rangle"] \\ \mathbb{N} \times \mathbb{N}^\omega \end{tikzcd}$$

$\mathsf{head}(n_0, n_1, n_2, \ldots) = n_0$

$\mathsf{tail}(n_0, n_1, n_2, \ldots) = (n_1, n_2, \ldots)$

Stream Differential Equations

$$\begin{array}{c} \mathbb{N}^\omega \\ \Big\downarrow \langle \mathsf{head}, \mathsf{tail} \rangle \\ \mathbb{N} \times \mathbb{N}^\omega \end{array}$$

Also here we shall write

$(n_0, n_1, n_2, \ldots)(0) = n_0$

$(n_0, n_1, n_2, \ldots)' = (n_1, n_2, n_3, \ldots)$

Finality of streams

$$\begin{tikzcd} X \arrow[r, dashed, "\exists! f"] \arrow[d, "\langle \mathrm{out}, \mathrm{tr} \rangle"'] & \mathbb{N}^\omega \arrow[d, "\langle \mathrm{head}, \mathrm{tail} \rangle"] \\ \mathbb{N} \times X \arrow[r, dashed] & \mathbb{N} \times \mathbb{N}^\omega \end{tikzcd}$$

The function f, defined by

$$f(x) = (\mathrm{out}(x),\ \mathrm{out}(\mathrm{tr}(x)),\ \mathrm{out}(\mathrm{tr}(\mathrm{tr}(x))),\ \dots)$$

is the *unique* function making the diagram commute.

Solutions by finality

$$\begin{tikzcd} X \arrow[r, dashed, "\exists!\, f"] \arrow[d, "\langle \mathrm{out}, \mathrm{tr} \rangle"'] & \mathbb{N}^\omega \arrow[d, "\langle \mathrm{head}, \mathrm{tail} \rangle"] \\ \mathbb{N} \times X \arrow[r, dashed] & \mathbb{N} \times \mathbb{N}^\omega \end{tikzcd}$$

System of SDEs:

$$\{\, x(0) = \mathrm{out}(x) \quad \text{and} \quad x' = \mathrm{tr}(x) \,\}_{x \in X}$$

The (unique) **solution** is given by the collection of streams:

$$\{\, f(x) \,\}_{x \in X}$$

These streams **are** a solution of the SDEs, since

$$f(x)(0) = \mathrm{out}(x) \quad \text{and} \quad f(x)' = \mathrm{tr}(x)$$

Solutions by finality

$$\begin{tikzcd} X \arrow[r, dashed, "\exists!\, f"] \arrow[d, "\langle \mathrm{out}, \mathrm{tr} \rangle"'] & \mathbb{N}^\omega \arrow[d, "\langle \mathrm{head}, \mathrm{tail} \rangle"] \\ \mathbb{N} \times X \arrow[r, dashed] & \mathbb{N} \times \mathbb{N}^\omega \end{tikzcd}$$

System of SDEs:

$$\{\, x(0) = \mathrm{out}(x) \quad \text{and} \quad x' = \mathrm{tr}(x) \,\}_{x \in X}$$

The (unique) **solution** is given by the collection of streams:

$$\{\, f(x) \,\}_{x \in X}$$

These streams **are** a solution of the SDEs, since

$$f(x)(0) = \mathrm{out}(x) \quad \text{and} \quad f(x)' = \mathrm{tr}(x)$$

Solutions by finality

$$\begin{tikzcd}
X \arrow[r, dashed, "\exists!\, f"] \arrow[d, "\langle \mathrm{out}, \mathrm{tr} \rangle"'] & \mathbb{N}^\omega \arrow[d, "\langle \mathrm{head}, \mathrm{tail} \rangle"] \\
\mathbb{N} \times X \arrow[r, dashed] & \mathbb{N} \times \mathbb{N}^\omega
\end{tikzcd}$$

System of SDEs:

$$\{\, x(0) = \mathrm{out}(x) \quad \text{and} \quad x' = \mathrm{tr}(x) \,\}_{x \in X}$$

The (unique) **solution** is given by the collection of streams:

$$\{\, f(x) \,\}_{x \in X}$$

These streams **are** a solution of the SDEs, since

$$f(x)(0) = \mathrm{out}(x) \quad \text{and} \quad f(x)' = \mathrm{tr}(x)$$

Stream calculus is easy ...

... since **any** system of SDEs

$$\begin{tikzcd} X \arrow[d, "{\langle \mathrm{out}, \mathrm{tr} \rangle}"'] \\ \mathbb{N} \times X \end{tikzcd} \qquad \{\, x(0) = \mathrm{out}(x) \quad \text{and} \quad x' = \mathrm{tr}(x) \,\}_{x \in X}$$

has a (unique solution)

$$\{\, f(x) \,\}_{x \in X}$$

given by finality:

$$\begin{tikzcd} X \arrow[r, dashed, "\exists!\, f"] \arrow[d, "{\langle \mathrm{out}, \mathrm{tr} \rangle}"'] & \mathbb{N}^\omega \arrow[d, "{\langle \mathrm{head}, \mathrm{tail} \rangle}"] \\ \mathbb{N} \times X \arrow[r, dashed] & \mathbb{N} \times \mathbb{N}^\omega \end{tikzcd}$$

Stream calculus is easy . . .

. . . since **any** system of SDEs

$$\begin{tikzcd} X \arrow[d, "\langle \mathrm{out}, \mathrm{tr} \rangle"'] \\ \mathbb{N} \times X \end{tikzcd} \qquad \{\, x(0) = \mathrm{out}(x) \quad \text{and} \quad x' = \mathrm{tr}(x) \,\}_{x \in X}$$

has a (unique solution)

$$\{\, f(x) \,\}_{x \in X}$$

given by finality:

$$\begin{tikzcd} X \arrow[r, dashed, "\exists ! f"] \arrow[d, "\langle \mathrm{out}, \mathrm{tr} \rangle"'] & \mathbb{N}^\omega \arrow[d, "\langle \mathrm{head}, \mathrm{tail} \rangle"] \\ \mathbb{N} \times X \arrow[r, dashed] & \mathbb{N} \times \mathbb{N}^\omega \end{tikzcd}$$

Stream calculus is easy . . .

. . . since **any** system of SDEs

$$\begin{array}{c} X \\ \langle \mathrm{out}, \mathrm{tr} \rangle \Big\downarrow \\ \mathbb{N} \times X \end{array} \qquad \{\, x(0) = \mathrm{out}(x) \quad \text{and} \quad x' = \mathrm{tr}(x) \,\}_{x \in X}$$

has a (unique solution)

$$\{\, f(x) \,\}_{x \in X}$$

given by finality:

$$\begin{tikzcd} X \arrow[r, dashed, "\exists!\, f"] \arrow[d, "\langle \mathrm{out}, \mathrm{tr} \rangle"'] & \mathbb{N}^{\omega} \arrow[d, "\langle \mathrm{head}, \mathrm{tail} \rangle"] \\ \mathbb{N} \times X \arrow[r, dashed] & \mathbb{N} \times \mathbb{N}^{\omega} \end{tikzcd}$$

Example

$$
\begin{tikzcd}
\{x, y\} \arrow[r, dashed, "\exists!\, f"] \arrow[d, "\langle \mathrm{out}, \mathrm{tr} \rangle"'] & \mathbb{N}^\omega \arrow[d] \\
\mathbb{N} \times \{x, y\} \arrow[r, dashed] & \mathbb{N} \times \mathbb{N}^\omega
\end{tikzcd}
$$

SDEs: $x(0) = 0, \quad x' = y$ and $y(0) = 1, \quad y' = x$

Solution: $f(x) = (0, 1, 0, 1, \dots), \quad f(y) = (1, 0, 1, 0, \dots)$

Example: infinite system of SDEs

$$
\begin{tikzcd}
\mathbb{N}^\omega \times \mathbb{N}^\omega \arrow[rr, dashed, "\exists!\, f"] \arrow[d, "\langle \mathrm{out}, \mathrm{tr} \rangle"'] & & \mathbb{N}^\omega \arrow[d] \\
\mathbb{N} \times (\mathbb{N}^\omega \times \mathbb{N}^\omega) \arrow[rr, dashed] & & \mathbb{N} \times \mathbb{N}^\omega
\end{tikzcd}
$$

SDEs:

$$(\sigma, \tau)(0) = \sigma(0) + \tau(0), \qquad (\sigma, \tau)' = (\sigma', \tau') \qquad (\forall\, \sigma, \tau \in \mathbb{N}^\omega\,)$$

Solution:

$$f(\sigma, \tau) = (\sigma(0) + \tau(0),\ \sigma(1) + \tau(1),\ \ldots)$$

Example: infinite system of SDEs

$$\begin{tikzcd}
\mathbb{N}^\omega \times \mathbb{N}^\omega \arrow[r, dashed, "\exists!\, +"] \arrow[d] & \mathbb{N}^\omega \arrow[d] \\
\mathbb{N} \times (\mathbb{N}^\omega \times \mathbb{N}^\omega) \arrow[r, dashed] & \mathbb{N} \times \mathbb{N}^\omega
\end{tikzcd}$$

SDEs:

$$(\sigma + \tau)(0) = \sigma(0) + \tau(0), \qquad (\sigma + \tau)' = \sigma' + \tau' \qquad (\forall\, \sigma, \tau \in \mathbb{N}^\omega\,)$$

Solution:

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

Example: infinite system of SDEs

$$
\begin{tikzcd}
\mathbb{N}^\omega \times \mathbb{N}^\omega \arrow[r, dashed, "\exists!\, +"] \arrow[d] & \mathbb{N}^\omega \arrow[d] \\
\mathbb{N} \times (\mathbb{N}^\omega \times \mathbb{N}^\omega) \arrow[r, dashed] & \mathbb{N} \times \mathbb{N}^\omega
\end{tikzcd}
$$

SDEs:

$$(\sigma+\tau)(0) = \sigma(0)+\tau(0), \qquad (\sigma+\tau)' = \sigma'+\tau' \qquad (\forall\, \sigma, \tau \in \mathbb{N}^\omega\,)$$

Solution:
This formula is not really relevant. SDE says it all.

Example: in the end . . .

. . . we simply will say: Let the function

$$+ : \mathbb{N}^\omega \times \mathbb{N}^\omega \to \mathbb{N}^\omega$$

be given by the following system of SDEs:

$$(\sigma + \tau)(0) = \sigma(0) + \tau(0), \qquad (\sigma + \tau)' = \sigma' + \tau' \qquad (\forall\, \sigma, \tau \in \mathbb{N}^\omega\,)$$

Example: shuffle product

Let the function

$$\otimes : \mathbb{N}^\omega \times \mathbb{N}^\omega \to \mathbb{N}^\omega$$

be given by the following system of SDEs:

$$(\sigma \otimes \tau)(0) = \sigma(0)\tau(0), \qquad (\sigma \otimes \tau)' = (\sigma' \otimes \tau) + (\sigma \otimes \tau')$$

Solution: $\quad (\sigma \otimes \tau)(n) = \sum_{k=0}^{n} \binom{n}{k} \cdot \sigma(k) \cdot \tau(n-k)$

Example: shuffle product

Let the function

$$\otimes : \mathbb{N}^\omega \times \mathbb{N}^\omega \to \mathbb{N}^\omega$$

be given by the following system of SDEs:

$$(\sigma \otimes \tau)(0) = \sigma(0)\tau(0), \qquad (\sigma \otimes \tau)' = (\sigma' \otimes \tau) + (\sigma \otimes \tau')$$

Again: this formula is not important. SDE says it all.

Proofs by coinduction

$R \subseteq \mathbb{N}^\omega \times \mathbb{N}^\omega$ is a stream bisimulation if

$$\forall (\sigma, \tau) \in R: \quad (i)\ \sigma(0) = \tau(0) \text{ and } (ii)\ (\sigma', \tau') \in R$$

Theorem [Coinduction proof principle]:

$$(\sigma, \tau) \in R \quad \Rightarrow \quad \sigma = \tau$$

Proof: exercise.

Proofs by coinduction

$R \subseteq \mathbb{N}^\omega \times \mathbb{N}^\omega$ is a stream bisimulation if

$$\forall(\sigma, \tau) \in R: \quad (i)\ \ \sigma(0) = \tau(0) \ \text{ and } \quad (ii)\ \ (\sigma', \tau') \in R$$

Theorem [Coinduction proof principle]:

$$(\sigma, \tau) \in R \quad \Rightarrow \quad \sigma = \tau$$

Proof: exercise.

Coinduction: example

For all $\sigma, \tau, \rho \in \mathbb{N}^\omega$:

$$(\sigma \otimes \tau) \otimes \rho = \sigma \otimes (\tau \otimes \rho)$$

Proof:

$$R = \{ ((\sigma \otimes \tau) \otimes \rho,\ \sigma \otimes (\tau \otimes \rho)) \mid \sigma, \tau, \rho \in \mathbb{N}^\omega \}$$

is a stream bisimulation relation *up-to* $+$.

Coinduction: example

For all $\sigma, \tau, \rho \in \mathbb{N}^\omega$:

$$(\sigma \otimes \tau) \otimes \rho = \sigma \otimes (\tau \otimes \rho)$$

Proof:

$$\textit{R} = \{ ((\sigma \otimes \tau) \otimes \rho,\ \sigma \otimes (\tau \otimes \rho)) \mid \sigma, \tau, \rho \in \mathbb{N}^\omega \}$$

is a stream bisimulation relation *up-to* $+$, since

$$((\sigma \otimes \tau) \otimes \rho)' = (\sigma' \otimes \tau) \otimes \rho + (\sigma \otimes \tau') \otimes \rho + (\sigma \otimes \tau) \otimes \rho'$$

$$(\sigma \otimes (\tau \otimes \rho))' = \sigma' \otimes (\tau \otimes \rho) + \sigma \otimes (\tau' \otimes \rho) + \sigma \otimes (\tau \otimes \rho')$$

Coinduction: example

For all $\sigma, \tau, \rho \in \mathbb{N}^\omega$:

$$(\sigma \otimes \tau) \otimes \rho = \sigma \otimes (\tau \otimes \rho)$$

Exercise: try and give a proof using the formula

$$(\sigma \otimes \tau)(n) = \sum_{k=0}^{n} \binom{n}{k} \cdot \sigma(k) \cdot \tau(n-k)$$

Coinduction-up-to

Cf. Milner, Sangiorgi

Coinduction-up-to really is: Algebra + Coalgebra

Cf. Coalgebraic bisimulation-up-to
J. Rot, M. Bonsangue, and J. Rutten
LNCS 7741, 2013

Cf. Hacking nondeterminism with induction and coinduction
Filippo Bonchi and Damien Pous
Commun. ACM Vol. 58(2), 2015

More in Lecture four.

Coinduction-up-to

Cf. Milner, Sangiorgi

Coinduction-up-to really is: Algebra + Coalgebra

Cf. Coalgebraic bisimulation-up-to
J. Rot, M. Bonsangue, and J. Rutten
LNCS 7741, 2013

Cf. Hacking nondeterminism with induction and coinduction
Filippo Bonchi and Damien Pous
Commun. ACM Vol. 58(2), 2015

More in Lecture four.

2. Solving systems of SDEs

Previous definition of SDEs: **semantical**.

Next: **syntax**.

- Given: a **syntactically** presented system of SDEs.
- Goal: find its **solution**.
- Answer: use the **syntactic method** to construct a suitable stream coalgebra.
- Use **finality** (as before) to get the solution.

2. Solving systems of SDEs

Previous definition of SDEs: **semantical.**

Next: **syntax.**

- Given: a **syntactically** presented system of SDEs.
- Goal: find its **solution.**
- Answer: use the **syntactic method** to construct a suitable stream coalgebra.
- Use **finality** (as before) to get the solution.

2. Solving systems of SDEs

Previous definition of SDEs: **semantical.**

Next: **syntax.**

- Given: a **syntactically** presented system of SDEs.
- Goal: find its **solution.**
- Answer: use the **syntactic method** to construct a suitable stream coalgebra.
- Use **finality** (as before) to get the solution.

Examples

The SDE:

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

defines

$$\sigma = (1, 1, 1, \ldots)$$

The SDE:

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

defines the Fibonacci numbers:

$$\sigma = (1, 1, 2, 3, 5, 8, \ldots)$$

Examples

The SDE:

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

defines

$$\sigma = (1, 1, 1, \ldots)$$

The SDE:

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

defines the Fibonacci numbers:

$$\sigma = (1, 1, 2, 3, 5, 8, \ldots)$$

Examples

The SDE:

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

defines

$$\sigma = (1, 1, 1, \ldots)$$

The SDE:

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

defines the Fibonacci numbers:

$$\sigma = (1, 1, 2, 3, 5, 8, \ldots)$$

Examples

The SDE:

$$(\sigma + \tau)' = \sigma' + \tau' \qquad (\sigma + \tau)(0) = \sigma(0) + \tau(0)$$

defines pointwise sum:

$$(\sigma + \tau)(n) = \sigma(n) + \tau(n)$$

The SDE:

$$(\sigma \times \tau)' = (\sigma' \times \tau) + ([\sigma(0)] \times \tau') \qquad (\sigma \times \tau)(0) = \sigma(0) \cdot \tau(0)$$

(where $[\sigma(0)] = (\sigma(0), 0, 0, 0, \ldots)$) defines convolution product:

$$(\sigma \times \tau)(n) = \sum_{k=0}^{n} \sigma(k) \cdot \tau(n-k)$$

Examples

The SDE:

$$(\sigma + \tau)' = \sigma' + \tau' \qquad (\sigma + \tau)(0) = \sigma(0) + \tau(0)$$

defines pointwise sum:

$$(\sigma + \tau)(n) = \sigma(n) + \tau(n)$$

The SDE:

$$(\sigma \times \tau)' = (\sigma' \times \tau) + ([\sigma(0)] \times \tau') \qquad (\sigma \times \tau)(0) = \sigma(0) \cdot \tau(0)$$

(where $[\sigma(0)] = (\sigma(0), 0, 0, 0, \ldots)$) defines convolution product:

$$(\sigma \times \tau)(n) = \sum_{k=0}^{n} \sigma(k) \cdot \tau(n-k)$$

Examples

The SDE:

$$(\sigma + \tau)' = \sigma' + \tau' \qquad (\sigma + \tau)(0) = \sigma(0) + \tau(0)$$

defines pointwise sum:

$$(\sigma + \tau)(n) = \sigma(n) + \tau(n)$$

The SDE:

$$(\sigma \times \tau)' = (\sigma' \times \tau) + ([\sigma(0)] \times \tau') \qquad (\sigma \times \tau)(0) = \sigma(0) \cdot \tau(0)$$

(where $[\sigma(0)] = (\sigma(0), 0, 0, 0, \ldots)$) defines convolution product:

$$(\sigma \times \tau)(n) = \sum_{k=0}^{n} \sigma(k) \cdot \tau(n-k)$$

Examples

The SDE:

$$(\sigma + \tau)' = \sigma' + \tau' \qquad (\sigma + \tau)(0) = \sigma(0) + \tau(0)$$

defines pointwise sum:

$$(\sigma + \tau)(n) = \sigma(n) + \tau(n)$$

The SDE:

$$(\sigma \times \tau)' = (\sigma' \times \tau) + ([\sigma(0)] \times \tau') \qquad (\sigma \times \tau)(0) = \sigma(0) \cdot \tau(0)$$

(where $[\sigma(0)] = (\sigma(0), 0, 0, 0, \ldots)$) defines convolution product:

$$(\sigma \times \tau)(n) = \sum_{k=0}^{n} \sigma(k) \cdot \tau(n-k)$$

The syntactic method

A general method for solving systems of SDEs.

It works for a fairly large class of systems of SDEs.

We explain it by means of an example: the Hamming numbers.

The syntactic method

A general method for solving systems of SDEs.

It works for a fairly large class of systems of SDEs.

We explain it by means of an example: the Hamming numbers.

The Hamming numbers

Cf. Dijkstra's [EDW792].

All natural numbers, in increasing order, that have no other prime factors than 2 and 3 (and 5):

$$\begin{aligned} \gamma &= (2^0 3^0,\, 2^1 3^0,\, 2^0 3^1,\, 2^2 3^0,\, 2^1 3^1,\, 2^3 3^0,\, 2^0 3^2,\, 2^2 3^1,\, \ldots) \\ &= (1, 2, 3, 4, 6, 8, 9, 12, \ldots) \end{aligned}$$

We define γ by the stream differential equation

$$\gamma' = (2 \times \gamma) \parallel (3 \times \gamma) \qquad \gamma(0) = 1$$

Note: this is not classical mathematics.

The Hamming numbers

Cf. Dijkstra's [EDW792].

All natural numbers, in increasing order, that have no other prime factors than 2 and 3 (and 5):

$$
\begin{aligned}
\gamma &= (2^0 3^0,\, 2^1 3^0,\, 2^0 3^1,\, 2^2 3^0,\, 2^1 3^1,\, 2^3 3^0,\, 2^0 3^2,\, 2^2 3^1,\, \ldots) \\
&= (1, 2, 3, 4, 6, 8, 9, 12, \ldots)
\end{aligned}
$$

We define γ by the stream differential equation

$$\gamma' = (2 \times \gamma) \parallel (3 \times \gamma) \qquad \gamma(0) = 1$$

Note: this is not classical mathematics.

The stream differential equation

$$\gamma' = (2 \times \gamma) \parallel (3 \times \gamma) \qquad \gamma(0) = 1$$

Here the *ordered merge* $\parallel : \mathbb{N}^\omega \times \mathbb{N}^\omega \to \mathbb{N}^\omega$ is defined by

$$(\sigma \parallel \tau)' = \begin{cases} \sigma' \parallel \tau & \text{if } \sigma(0) < \tau(0) \\ \sigma' \parallel \tau' & \text{if } \sigma(0) = \tau(0) \\ \sigma \parallel \tau' & \text{if } \sigma(0) > \tau(0) \end{cases}$$

$$(\sigma \parallel \tau)(0) = \begin{cases} \sigma(0) & \text{if } \sigma(0) < \tau(0) \\ \tau(0) & \text{if } \sigma(0) \geq \tau(0) \end{cases}$$

and $2 \times \sigma$ (and similarly $3 \times \sigma$) is defined by

$$(2 \times \sigma)' = 2 \times (\sigma') \qquad (2 \times \sigma)(0) = 2 \cdot \sigma(0)$$

Syntactic solution method

Goal: to prove the *unique existence* of a solution for

$$\gamma' = (2 \times \gamma) \parallel (3 \times \gamma) \qquad \gamma(0) = 1$$

Assuming the solution exists, we compute the first few derivatives of γ:

$$\begin{aligned}
\gamma^{(1)} &= (2 \times \gamma) \parallel (3 \times \gamma) \\
\gamma^{(2)} &= (2 \times ((2 \times \gamma) \parallel (3 \times \gamma))) \parallel (3 \times \gamma) \\
\gamma^{(3)} &= (2 \times ((2 \times \gamma) \parallel (3 \times \gamma))) \parallel (3 \times ((2 \times \gamma) \parallel (3 \times \gamma)))
\end{aligned}$$

The idea: define **syntactic** terms for all possible such righthand sides.

Syntactic solution method

Goal: to prove the *unique existence* of a solution for

$$\gamma' = (2 \times \gamma) \parallel (3 \times \gamma) \qquad \gamma(0) = 1$$

Assuming the solution exists, we compute the first few derivatives of γ:

$$\begin{aligned}
\gamma^{(1)} &= (2 \times \gamma) \parallel (3 \times \gamma) \\
\gamma^{(2)} &= (2 \times ((2 \times \gamma) \parallel (3 \times \gamma))) \parallel (3 \times \gamma) \\
\gamma^{(3)} &= (2 \times ((2 \times \gamma) \parallel (3 \times \gamma))) \parallel (3 \times ((2 \times \gamma) \parallel (3 \times \gamma)))
\end{aligned}$$

The idea: define **syntactic** terms for all possible such righthand sides.

Syntactic solution method

Goal: to prove the *unique existence* of a solution for

$$\gamma' = (2 \times \gamma) \parallel (3 \times \gamma) \qquad \gamma(0) = 1$$

Assuming the solution exists, we compute the first few derivatives of γ:

$$\begin{aligned}
\gamma^{(1)} &= (2 \times \gamma) \parallel (3 \times \gamma) \\
\gamma^{(2)} &= (2 \times ((2 \times \gamma) \parallel (3 \times \gamma))) \parallel (3 \times \gamma) \\
\gamma^{(3)} &= (2 \times ((2 \times \gamma) \parallel (3 \times \gamma))) \parallel (3 \times ((2 \times \gamma) \parallel (3 \times \gamma)))
\end{aligned}$$

The idea: define **syntactic** terms for all possible such righthand sides.

The term coalgebra

Term $\ni$ $t ::=$ c $|$ $\underline{\sigma}$ $(\sigma \in \mathbb{N}^\omega)$ $|$ 2times(t) $|$ 3times(t) $|$ merge(t_1, t_2)

Next we turn the set Term into a stream coalgebra

$$\text{Term} \xrightarrow{\langle \text{out}, \text{tr} \rangle} \mathbb{N} \times \text{Term}$$

by defining functions out : Term $\to \mathbb{N}$ and tr : Term $\to$ Term by *induction* on the structure of terms, following the stream diff. eqn's.

The term coalgebra

$$\mathsf{Term} \ni \; t ::= \; \mathsf{c} \mid \underline{\sigma} \; (\sigma \in \mathbb{N}^{\omega}) \mid \mathsf{2times}(t) \mid \mathsf{3times}(t) \mid \mathsf{merge}(t_1, t_2)$$

Next we turn the set Term into a stream coalgebra

$$\mathsf{Term} \xrightarrow{\langle \mathsf{out}, \mathsf{tr} \rangle} \mathbb{N} \times \mathsf{Term}$$

by defining functions $\mathsf{out} : \mathsf{Term} \to \mathbb{N}$ and $\mathsf{tr} : \mathsf{Term} \to \mathsf{Term}$ by *induction* on the structure of terms, following the stream diff. eqn's.

The solution

By finality,

$$\begin{tikzcd} \mathsf{Term} \arrow[r, dashed, "\exists\,!\, f"] \arrow[d, "\langle \mathsf{out}, \mathsf{tr} \rangle"'] & \mathbb{N}^\omega \arrow[d] \\ \mathbb{N} \times \mathsf{Term} \arrow[r] & \mathbb{N} \times \mathbb{N}^\omega \end{tikzcd}$$

Using f, we define

$$\begin{aligned} \gamma &= f(c) \\ \sigma \parallel \tau &= f(\mathsf{merge}(\underline{\sigma}, \underline{\tau})) \end{aligned}$$

(and similarly for $2 \times \sigma$ and $3 \times \sigma$).

Finally one shows that, indeed,

$$\gamma' = (2 \times \gamma) \parallel (3 \times \gamma) \qquad \gamma(0) = 1$$

The solution

By finality,

$$\begin{tikzcd}
\mathsf{Term} \arrow[r, dashed, "\exists!\, f"] \arrow[d, "\langle \mathsf{out}, \mathsf{tr} \rangle"'] & \mathbb{N}^\omega \arrow[d] \\
\mathbb{N} \times \mathsf{Term} \arrow[r] & \mathbb{N} \times \mathbb{N}^\omega
\end{tikzcd}$$

Using f, we define

$$\begin{aligned} \gamma &= f(c) \\ \sigma \parallel \tau &= f(\mathsf{merge}(\underline{\sigma}, \underline{\tau})) \end{aligned}$$

(and similarly for $2 \times \sigma$ and $3 \times \sigma$).

Finally one shows that, indeed,

$$\gamma' = (2 \times \gamma) \parallel (3 \times \gamma) \qquad \gamma(0) = 1$$

The solution

By finality,

$$\begin{tikzcd}
\mathsf{Term} \arrow[r, dashed, "\exists!\, f"] \arrow[d, "\langle \mathsf{out}, \mathsf{tr} \rangle"'] & \mathbb{N}^\omega \arrow[d] \\
\mathbb{N} \times \mathsf{Term} \arrow[r] & \mathbb{N} \times \mathbb{N}^\omega
\end{tikzcd}$$

Using f, we define

$$\begin{aligned} \gamma &= f(c) \\ \sigma \parallel \tau &= f(\mathsf{merge}(\underline{\sigma}, \underline{\tau})) \end{aligned}$$

(and similarly for $2 \times \sigma$ and $3 \times \sigma$).

Finally one shows that, indeed,

$$\gamma' = (2 \times \gamma) \parallel (3 \times \gamma) \qquad \gamma(0) = 1$$

Not all is well

Let the function

$$\text{even} : \mathbb{N}^\omega \to \mathbb{N}^\omega$$

be given by the following system of SDEs:

$$(\,\text{even}(\sigma)\,)(0) = \sigma(0), \quad \text{even}(\sigma)' = \text{even}(\sigma'')$$

(Solution: $\text{even}(\sigma) = (\sigma(0), \sigma(2), \sigma(4), \ldots)$.)

Not all is well

Let the function

$$\text{even} : \mathbb{N}^\omega \to \mathbb{N}^\omega$$

be given by the following system of SDEs:

$$(\,\text{even}(\sigma)\,)(0) = \sigma(0), \quad \text{even}(\sigma)' = \text{even}(\sigma'')$$

(Solution: $\text{even}(\sigma) = (\sigma(0), \sigma(2), \sigma(4), \ldots)$.)

Not all is well

Now consider the following SDE:

$$x(0) = 0 \qquad x' = \mathrm{even}(x)$$

It has **many** solutions, such as

$$x = (0, 0, 0, \ldots) \quad x = (0, 0, 1, 1, 1, \ldots)$$

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

Exercise: how many solutions are there?

Not all is well

Now consider the following SDE:

$$x(0) = 0 \qquad x' = \text{even}(x)$$

It has **many** solutions, such as

$$x = (0, 0, 0, \ldots) \quad x = (0, 0, 1, 1, 1, \ldots)$$

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

Exercise: how many solutions are there?

Not all is well

Now consider the following SDE:

$$x(0) = 0 \qquad x' = \mathrm{even}(x)$$

It has **many** solutions, such as

$$x = (0, 0, 0, \ldots) \quad x = (0, 0, 1, 1, 1, \ldots)$$

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

Exercise: how many solutions are there?

The syntactic format is important

The syntactic method does not work for

$$x(0) = 0 \qquad x' = \text{even}(x)$$

The problem is that it does **not** translate uniquely to a corresponding stream coalgebra.

The technical problem is the second derivative in

$$\text{even}(\sigma)' = \text{even}(\sigma'')$$

The syntactic format is important

The syntactic method does not work for

$$x(0) = 0 \qquad x' = \text{even}(x)$$

The problem is that it does **not** translate uniquely to a corresponding stream coalgebra.

The technical problem is the second derivative in

$$\text{even}(\sigma)' = \text{even}(\sigma'')$$

The syntactic format is important

The syntactic method does not work for

$$x(0) = 0 \qquad x' = \mathrm{even}(x)$$

The problem is that it does **not** translate uniquely to a corresponding stream coalgebra.

The technical problem is the second derivative in

$$\mathrm{even}(\sigma)' = \mathrm{even}(\sigma'')$$

3. Formats for SDEs

- A general format for the syntactic method
- Three well-known sub-classes:
  - Periodic streams
  - Rational streams
  - Context-free streams
- (Cf. formal languages.)

A useful set of operators on $\mathbb{R}^\omega$

$$[r] = (r, 0, 0, 0, \ldots) \qquad \text{for each } r \in \mathbb{R}$$

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

$$(\sigma + \tau)(n) = \sigma(n) + \tau(n)$$

$$(\sigma \times \tau)(n) = \sum_{k=0}^{n} \sigma(k) \cdot \tau(n-k)$$

$$\sigma \times \sigma^{-1} = [1] \qquad (\sigma(0) \neq 0)$$

The corresponding system of SDEs

derivative:	*initial value:*
$[r]' = [0]$	$[r](0) = r$
$X' = [1]$	$X(0) = 0$
$(\sigma + \tau)' = \sigma' + \tau'$	$(\sigma + \tau)(0) = \sigma(0) + \tau(0)$
$(\sigma \times \tau)' = (\sigma' \times \tau) + ([\sigma(0)] \times \tau')$	$(\sigma \times \tau)(0) = \sigma(0) \cdot \tau(0)$
$(\sigma^{-1})' = -[\sigma(0)^{-1}] \times \sigma' \times \sigma^{-1}$	$(\sigma^{-1})(0) = \sigma(0)^{-1}$

Illustrating the format for our syntactic method

derivative:	*initial value:*
$[r]' = [0]$	$[r](0) = r$
$X' = [1]$	$X(0) = 0$
$(\sigma + \tau)' = \sigma' + \tau'$	$(\sigma + \tau)(0) = \sigma(0) + \tau(0)$
$(\sigma \times \tau)' = (\sigma' \times \tau) + ([\sigma(0)] \times \tau')$	$(\sigma \times \tau)(0) = \sigma(0) \cdot \tau(0)$
$(\sigma^{-1})' = -[\sigma(0)^{-1}] \times \sigma' \times \sigma^{-1}$	$(\sigma^{-1})(0) = \sigma(0)^{-1}$

The syntactic method applies in general to **this kind** of SDEs.

We shall explain **“this kind”**.

Illustrating the format for our syntactic method

derivative:	*initial value:*
$[r]' = [0]$	$[r](0) = r$
$X' = [1]$	$X(0) = 0$
$(\sigma + \tau)' = \sigma' + \tau'$	$(\sigma + \tau)(0) = \sigma(0) + \tau(0)$
$(\sigma \times \tau)' = (\sigma' \times \tau) + ([\sigma(0)] \times \tau')$	$(\sigma \times \tau)(0) = \sigma(0) \cdot \tau(0)$
$(\sigma^{-1})' = -[\sigma(0)^{-1}] \times \sigma' \times \sigma^{-1}$	$(\sigma^{-1})(0) = \sigma(0)^{-1}$

The syntactic method applies in general to **this kind** of SDEs.

We shall explain **“this kind”**.

Illustrating the format for our syntactic method

derivative:
$[r]' = [0]$
$X' = [1]$
$(\sigma + \tau)' = \sigma' + \tau'$
$(\sigma \times \tau)' = (\sigma' \times \tau) + ([\sigma(0)] \times \tau')$
$(\sigma^{-1})' = -[\sigma(0)^{-1}] \times \sigma' \times \sigma^{-1}$

On the left: terms with one operator (possibly a constant) ...

Illustrating the format for our syntactic method

derivative:
$[r]' = [0]$
$X' = [1]$
$(\sigma + \tau)' = \sigma' + \tau'$
$(\sigma \times \tau)' = (\sigma' \times \tau) + ([\sigma(0)] \times \tau')$
$(\sigma^{-1})' = -[\sigma(0)^{-1}] \times \sigma' \times \sigma^{-1}$

On the left: . . . and stream variables.

Illustrating the format for our syntactic method

derivative:
$[r]' = [0]$
$X' = [1]$
$(\sigma + \tau)' = \sigma' + \tau'$
$(\sigma \times \tau)' = (\sigma' \times \tau) + ([\sigma(0)] \times \tau')$
$(\sigma^{-1})' = -[\sigma(0)^{-1}] \times \sigma' \times \sigma^{-1}$

On the right: terms built from various operators . . .

Illustrating the format for our syntactic method

derivative:
$[r]' = [0]$ $X' = [1]$ $(\sigma + \tau)' = \sigma' + \tau'$ $(\sigma \times \tau)' = (\sigma' \times \tau) + ([\sigma(0)] \times \tau')$ $(\sigma^{-1})' = -[\sigma(0)^{-1}] \times \sigma' \times \sigma^{-1}$

On the right: . . . and stream variables . . .

Illustrating the format for our syntactic method

derivative:
$[r]' = [0]$
$X' = [1]$
$(\sigma + \tau)' = \sigma' + \tau'$
$(\sigma \times \tau)' = (\sigma' \times \tau) + ([\sigma(0)] \times \tau')$
$(\sigma^{-1})' = -[\sigma(0)^{-1}] \times \sigma' \times \sigma^{-1}$

On the right: ... and derivatives of stream variables ...

(**no** double derivatives)

Illustrating the format for our syntactic method

derivative:
$[r]' = [0]$ $X' = [1]$ $(\sigma + \tau)' = \sigma' + \tau'$ $(\sigma \times \tau)' = (\sigma' \times \tau) + ([\sigma(0)] \times \tau')$ $(\sigma^{-1})' = -[\sigma(0)^{-1}] \times \sigma' \times \sigma^{-1}$

On the right: ... and derivatives of stream variables ...

(**no** double derivatives)

Illustrating the format for our syntactic method

derivative:
$[r]' = [0]$ $X' = [1]$ $(\sigma + \tau)' = \sigma' + \tau'$ $(\sigma \times \tau)' = \sigma' \times \tau) + ([\sigma(0)] \times \tau')$ $(\sigma^{-1})' = -[\sigma(0)^{-1}] \times \sigma' \times \sigma^{-1}$

On the right: ... and initial values of stream variables.

The syntactic method

Theorem

Any system of SDEs such as

derivative:	*initial value:*
$[r]' = [0]$	$[r](0) = r$
$X' = [1]$	$X(0) = 0$
$(\sigma + \tau)' = \sigma' + \tau'$	$(\sigma + \tau)(0) = \sigma(0) + \tau(0)$
$(\sigma \times \tau)' = (\sigma' \times \tau) + ([\sigma(0)] \times \tau')$	$(\sigma \times \tau)(0) = \sigma(0) \cdot \tau(0)$
$(\sigma^{-1})' = -[\sigma(0)^{-1}] \times \sigma' \times \sigma^{-1}$	$(\sigma^{-1})(0) = \sigma(0)^{-1}$

has a unique solution.

Proof: By the syntactic method.

The syntactic method

Theorem

Any system of SDEs such as

derivative:	*initial value:*
$[r]' = [0]$	$[r](0) = r$
$X' = [1]$	$X(0) = 0$
$(\sigma + \tau)' = \sigma' + \tau'$	$(\sigma + \tau)(0) = \sigma(0) + \tau(0)$
$(\sigma \times \tau)' = (\sigma' \times \tau) + ([\sigma(0)] \times \tau')$	$(\sigma \times \tau)(0) = \sigma(0) \cdot \tau(0)$
$(\sigma^{-1})' = -[\sigma(0)^{-1}] \times \sigma' \times \sigma^{-1}$	$(\sigma^{-1})(0) = \sigma(0)^{-1}$

has a unique solution.

Proof: By the syntactic method.

Three well-known classes of streams

By **restricting** our format further, we obtain various concrete classes of streams.

We mention three of them:

- Periodic streams
- Rational streams
- Context-free streams

Three well-known classes of streams

By **restricting** our format further, we obtain various concrete classes of streams.

We mention three of them:

- Periodic streams
- Rational streams
- Context-free streams

Three well-known classes of streams

initial value	derivative	solution
$\sigma(0) = 1$	$\sigma' = \sigma$	$(1, 1, 1, \ldots)$
$\sigma(0) = 1$	$\sigma' = \sigma + \sigma$	$(2^0, 2^1, 2^2, \ldots)$
$\sigma(0) = 1$	$\sigma' = \sigma \times \sigma$	$(1, 1, 2, 5, 14, 42, \ldots)$
		Catalan numbers

Three well-known classes of streams

initial value	derivative	solution
$\sigma(0) = 1$	$\sigma' = \sigma$	$(1, 1, 1, \ldots)$
$\sigma(0) = 1$	$\sigma' = \sigma + \sigma$	$(2^0, 2^1, 2^2, \ldots)$
$\sigma(0) = 1$	$\sigma' = \sigma \times \sigma$	$(1, 1, 2, 5, 14, 42, \ldots)$
		Catalan numbers

Three well-known classes of streams

initial value	derivative	format righthand side
$\sigma(0) = 1$	$\sigma' = \sigma$	one stream variable
$\sigma(0) = 1$	$\sigma' = \sigma + \sigma$	also sums (and scalars)
$\sigma(0) = 1$	$\sigma' = \sigma \times \sigma$	also products

Three well-known classes of streams

initial value	derivative	format righthand side
$\sigma(0) = 1$	$\sigma' = \sigma$	one stream variable
$\sigma(0) = 1$	$\sigma' = \sigma + \sigma$	also sums (and scalars)
$\sigma(0) = 1$	$\sigma' = \sigma \times \sigma$	also products

Three well-known classes of streams

initial value	derivative	closed form for solution
$\sigma(0) = 1$	$\sigma' = \sigma$	1^ω
$\sigma(0) = 1$	$\sigma' = \sigma + \sigma$	$\frac{1}{1-2X}$
$\sigma(0) = 1$	$\sigma' = \sigma \times \sigma$	??

Three well-known classes of streams

initial value	derivative	closed form for solution
$\sigma(0) = 1$	$\sigma' = \sigma$	1^ω
$\sigma(0) = 1$	$\sigma' = \sigma + \sigma$	$\frac{1}{1-2X}$
$\sigma(0) = 1$	$\sigma' = \sigma \times \sigma$	??

Three well-known classes of streams

initial value	derivative	class
$\sigma(0) = 1$	$\sigma' = \sigma$	perodic
$\sigma(0) = 1$	$\sigma' = \sigma + \sigma$	rational
$\sigma(0) = 1$	$\sigma' = \sigma \times \sigma$	context-free

Three well-known classes of streams

initial value	derivative	class
$\sigma(0) = 1$	$\sigma' = \sigma$	perodic
$\sigma(0) = 1$	$\sigma' = \sigma + \sigma$	rational
$\sigma(0) = 1$	$\sigma' = \sigma \times \sigma$	context-free

4. Streams and coinduction

We saw an elementary example of coinduction (when proving the associativity of the shuffle product).

Time allowing, we will next illustrate the coinduction proof principle for streams with a non-trivial example.

4. Streams and coinduction

We saw an elementary example of coinduction (when proving the associativity of the shuffle product).

Time allowing, we will next illustrate the coinduction proof principle for streams with a non-trivial example.

A proof by coinduction: Moessner's theorem

- A. Moessner (1951), proof by O. Perron (1951) and I. Paasche (1952).
- Cf. Ralf Hinze: Scans and convolutions - a calculational proof of Moessner's theorem (Oxford University, 2010).
- Our proof: by coinduction (Niqui & R., 2011) . . .
- . . . is a student's exercise.
- Cf. the original proof: serious binomial coefficient manipulation!!

Moessner's theorem (k = 3)

nat	1	2	3	4	5	6	7	8	9	10	11	12	⋯
$Drop_3$	1	2		4	5		7	8		10	11	⋯	
Σ	1	3	7	12	19	27	37	48	⋯				
$Drop_2$	1		7		19		37		⋯				
Σ	1	8	27	64	⋯								
	=												
nat^3	1^3	2^3	3^3	4^3	⋯								

Moessner's theorem ($k = 3$)

nat	1	2	3	4	5	6	7	8	9	10	11	12	$\cdots$
$Drop_3$	1	2		4	5		7	8		10	11	$\cdots$	
Σ	1	3	7	12	19	27	37	48	$\cdots$				
$Drop_2$	1		7		19		37		$\cdots$				
Σ	1	8	27	64	$\cdots$								
	=												
nat^3	1^3	2^3	3^3	4^3	$\cdots$								

Moessner's theorem ($k = 3$)

nat	1	2	3	4	5	6	7	8	9	10	11	12	$\cdots$
$Drop_3$	1	2		4	5		7	8		10	11	$\cdots$	
Σ	1	3	7	12	19	27	37	48	$\cdots$				
$Drop_2$	1		7		19		37		$\cdots$				
Σ	1	8	27	64	$\cdots$								
	$=$												
nat^3	1^3	2^3	3^3	4^3	$\cdots$								

Moessner's theorem (k = 3)

nat	1	2	3	4	5	6	7	8	9	10	11	12	$\cdots$
$Drop_3$	1	2		4	5		7	8		10	11	$\cdots$	
Σ	1	3	7	12	19	27	37	48	$\cdots$				
$Drop_2$	1		7		19		37		$\cdots$				
Σ	1	8	27	64	$\cdots$								
	=												
nat^3	1^3	2^3	3^3	4^3	$\cdots$								

Moessner's theorem (k = 3)

nat	1	2	3	4	5	6	7	8	9	10	11	12	$\cdots$
$Drop_3$	1	2		4	5		7	8		10	11	$\cdots$	
Σ	1	3	7	12	19	27	37	48	$\cdots$				
$Drop_2$	1		7		19		37		$\cdots$				
Σ	1	8	27	64	$\cdots$								
	=												
nat^3	1^3	2^3	3^3	4^3	$\cdots$								

Moessner's theorem (k = 3)

nat	1	2	3	4	5	6	7	8	9	10	11	12	$\cdots$
$Drop_3$	1	2		4	5		7	8		10	11	$\cdots$	
Σ	1	3	7	12	19	27	37	48	$\cdots$				
$Drop_2$	1		7		19		37		$\cdots$				
Σ	1	8	27	64	$\cdots$								
	$=$												
nat^3	1^3	2^3	3^3	4^3	$\cdots$								

Moessner's theorem (k = 4)

nat	1	2	3	4	5	6	7	8	9	10	11	$\cdots$
$Drop_4$	1	2	3		5	6	7		9	10	11	$\cdots$
Σ	1	3	6	11	17	24	33	43	54	$\cdots$		
$Drop_3$	1	3		11	17		33	43		67	81	$\cdots$
Σ	1	4	15	32	65	108	175	$\cdots$				
$Drop_2$	1		15		65		175	$\cdots$				
Σ	1	16	81	256	$\cdots$							
	=	1^4	2^4	3^4	4^4	$\cdots$						

Moessner's theorem (k = 4)

nat	1	2	3	4	5	6	7	8	9	10	11	$\cdots$
Drop$_4$	1	2	3		5	6	7		9	10	11	$\cdots$
Σ	1	3	6	11	17	24	33	43	54	$\cdots$		
Drop$_3$	1	3		11	17		33	43		67	81	$\cdots$
Σ	1	4	15	32	65	108	175	$\cdots$				
Drop$_2$	1		15		65		175	$\cdots$				
Σ	1	16	81	256	$\cdots$							
	=	1^4	2^4	3^4	4^4	$\cdots$						

Moessner's theorem (k = 4)

nat	1	2	3	4	5	6	7	8	9	10	11	$\cdots$
$Drop_4$	1	2	3		5	6	7		9	10	11	$\cdots$
Σ	1	3	6	11	17	24	33	43	54	$\cdots$		
$Drop_3$	1	3		11	17		33	43		67	81	$\cdots$
Σ	1	4	15	32	65	108	175	$\cdots$				
$Drop_2$	1		15		65		175	$\cdots$				
Σ	1	16	81	256	$\cdots$							
	=	1^4	2^4	3^4	4^4	$\cdots$						

Moessner's theorem ($k = 4$)

nat	1	2	3	4	5	6	7	8	9	10	11	$\cdots$
$Drop_4$	1	2	3		5	6	7		9	10	11	$\cdots$
Σ	1	3	6	11	17	24	33	43	54	$\cdots$		
$Drop_3$	1	3		11	17		33	43		67	81	$\cdots$
Σ	1	4	15	32	65	108	175	$\cdots$				
$Drop_2$	1		15		65		175	$\cdots$				
Σ	1	16	81	256	$\cdots$							
	$=$	1^4	2^4	3^4	4^4	$\cdots$						

Moessner's theorem (k = 4)

nat	1	2	3	4	5	6	7	8	9	10	11	$\cdots$
$Drop_4$	1	2	3		5	6	7		9	10	11	$\cdots$
Σ	1	3	6	11	17	24	33	43	54	$\cdots$		
$Drop_3$	1	3		11	17		33	43		67	81	$\cdots$
Σ	1	4	15	32	65	108	175	$\cdots$				
$Drop_2$	1		15		65		175	$\cdots$				
Σ	1	16	81	256	$\cdots$							
	$=$	1^4	2^4	3^4	4^4	$\cdots$						

Moessner's theorem (k = 4)

nat	1	2	3	4	5	6	7	8	9	10	11	$\cdots$
Drop$_4$	1	2	3		5	6	7		9	10	11	$\cdots$
Σ	1	3	6	11	17	24	33	43	54	$\cdots$		
Drop$_3$	1	3		11	17		33	43		67	81	$\cdots$
Σ	1	4	15	32	65	108	175	$\cdots$				
Drop$_2$	1		15		65		175	$\cdots$				
Σ	1	16	81	256	$\cdots$							
	$=$	1^4	2^4	3^4	4^4	$\cdots$						

Moessner's theorem (k = 4)

nat	1	2	3	4	5	6	7	8	9	10	11	$\cdots$
$Drop_4$	1	2	3		5	6	7		9	10	11	$\cdots$
Σ	1	3	6	11	17	24	33	43	54	$\cdots$		
$Drop_3$	1	3		11	17		33	43		67	81	$\cdots$
Σ	1	4	15	32	65	108	175	$\cdots$				
$Drop_2$	1		15		65		175	$\cdots$				
Σ	1	16	81	256	$\cdots$							
	$=$	1^4	2^4	3^4	4^4	$\cdots$						

Moessner's theorem (k = 4)

nat	1	2	3	4	5	6	7	8	9	10	11	$\cdots$
$Drop_4$	1	2	3		5	6	7		9	10	11	$\cdots$
Σ	1	3	6	11	17	24	33	43	54	$\cdots$		
$Drop_3$	1	3		11	17		33	43		67	81	$\cdots$
Σ	1	4	15	32	65	108	175	$\cdots$				
$Drop_2$	1		15		65		175	$\cdots$				
Σ	1	16	81	256	$\cdots$							
	$=$	1^4	2^4	3^4	4^4	$\cdots$						

Moessner's theorem (k = 5)

nat	1	2	3	4	5	6	7	8	9	10	11	$\cdots$
Drop$_5$	1	2	3	4		6	7	8	9		11	$\cdots$
Σ	1	3	6	10	16	23	31	40	51	$\cdots$		
Drop$_4$	1	3	6		16	23	31		51	$\cdots$		
etc.							$\cdots$					
	$=$	1^5	2^5	3^5	4^5	$\cdots$						

Moessner's theorem (k = 5)

nat	1	2	3	4	5	6	7	8	9	10	11	$\cdots$
Drop$_5$	1	2	3	4		6	7	8	9		11	$\cdots$
Σ	1	3	6	10	16	23	31	40	51	$\cdots$		
Drop$_4$	1	3	6		16	23	31		51	$\cdots$		
etc.							$\cdots$					
	$=$	1^5	2^5	3^5	4^5	$\cdots$						

Moessner's theorem (k = 5)

nat	1	2	3	4	5	6	7	8	9	10	11	$\cdots$
$Drop_5$	1	2	3	4		6	7	8	9		11	$\cdots$
Σ	1	3	6	10	16	23	31	40	51	$\cdots$		
$Drop_4$	1	3	6		16	23	31		51	$\cdots$		
etc.							$\cdots$					
	$=$	1^5	2^5	3^5	4^5	$\cdots$						

Moessner's theorem (k = 5)

nat	1	2	3	4	5	6	7	8	9	10	11	$\cdots$
$Drop_5$	1	2	3	4		6	7	8	9		11	$\cdots$
Σ	1	3	6	10	16	23	31	40	51	$\cdots$		
$Drop_4$	1	3	6		16	23	31		51	$\cdots$		
etc.							$\cdots$					
	$=$	1^5	2^5	3^5	4^5	$\cdots$						

Moessner's theorem (k = 5)

nat	1	2	3	4	5	6	7	8	9	10	11	$\cdots$
$Drop_5$	1	2	3	4		6	7	8	9		11	$\cdots$
Σ	1	3	6	10	16	23	31	40	51	$\cdots$		
$Drop_4$	1	3	6		16	23	31		51	$\cdots$		
etc.							$\cdots$					
	$=$	1^5	2^5	3^5	4^5	$\cdots$						

Moessner's theorem (k = 5)

nat	1	2	3	4	5	6	7	8	9	10	11	$\cdots$
Drop$_5$	1	2	3	4		6	7	8	9		11	$\cdots$
Σ	1	3	6	10	16	23	31	40	51	$\cdots$		
Drop$_4$	1	3	6		16	23	31		51	$\cdots$		
etc.							$\dots$					

$$= \quad 1^5 \quad 2^5 \quad 3^5 \quad 4^5 \quad \cdots$$

Approach: use coinduction on streams

Coinduction proof principle for streams:

$$(\sigma, \tau) \in R, \text{ bisimulation relation} \quad \Rightarrow \quad \sigma = \tau$$

We formulate Moessner's theorem as an equality of two streams.

Next we shall prove that these streams **are** equal . . . .

. . . by showing that they **behave** the same.

That is, we show that they are related by a bisimulation.

Approach: use coinduction on streams

Coinduction proof principle for streams:

$$(\sigma, \tau) \in R, \text{ bisimulation relation} \quad \Rightarrow \quad \sigma = \tau$$

We formulate Moessner's theorem as an equality of two streams.

Next we shall prove that these streams **are** equal . ...

... by showing that they **behave** the same.

That is, we show that they are related by a bisimulation.

Approach: use coinduction on streams

Coinduction proof principle for streams:

$$(\sigma, \tau) \in R, \text{ bisimulation relation} \quad \Rightarrow \quad \sigma = \tau$$

We formulate Moessner's theorem as an equality of two streams.

Next we shall prove that these streams **are** equal . ...

... by showing that they **behave** the same.

That is, we show that they are related by a bisimulation.

Approach: use coinduction on streams

Coinduction proof principle for streams:

$$(\sigma, \tau) \in R, \text{ bisimulation relation} \quad \Rightarrow \quad \sigma = \tau$$

We formulate Moessner's theorem as an equality of two streams.

Next we shall prove that these streams **are** equal . $\ldots$

$\ldots$ by showing that they **behave** the same.

That is, we show that they are related by a bisimulation.

Formalising Moessner's theorem

$$\mathrm{nat}^3 = \Sigma \circ \mathit{Drop}_2 \circ \Sigma \circ \mathit{Drop}_3(\,\mathrm{nat}\,)$$

We will define all of the above ingredients using

stream differential equations

This will

- make the inherent **circularity** explicit, and
- help us contruct a suitable **bisimulation** relation!

Formalising Moessner's theorem

$$\text{nat}^3 = \Sigma \circ \mathit{Drop}_2 \circ \Sigma \circ \mathit{Drop}_3(\text{nat})$$

We will define all of the above ingredients using

stream differential equations

This will

- make the inherent **circularity** explicit, and
- help us contruct a suitable **bisimulation** relation!

Formalising Moessner's theorem

$$\mathsf{nat}^3 = \Sigma \circ \mathit{Drop}_2 \circ \Sigma \circ \mathit{Drop}_3(\mathsf{nat})$$

where $\mathsf{nat} = (1, 2, 3, \ldots)$ satisfies

$$\mathsf{nat}(0) = 1 \qquad \mathsf{nat}' = \mathsf{nat} + \mathsf{ones}$$

with $\mathsf{ones} = (1, 1, 1, \ldots)$; and

$$\mathsf{nat}^3 = (1^3, 2^3, 3^3, \ldots) = \mathsf{nat} \odot \mathsf{nat} \odot \mathsf{nat}$$

with

$$(\sigma \odot \tau)(0) = \sigma(0) \cdot \tau(0) \qquad (\sigma \odot \tau)' = \sigma' \odot \tau'$$

Formalising Moessner's theorem

$$\mathsf{nat}^3 = \Sigma \circ \mathit{Drop}_2 \circ \Sigma \circ \mathit{Drop}_3(\,\mathsf{nat}\,)$$

where $\mathsf{nat} = (1, 2, 3, \ldots)$ satisfies

$$\mathsf{nat}(0) = 1 \qquad \mathsf{nat}' = \mathsf{nat} + \mathsf{ones}$$

with $\mathsf{ones} = (1, 1, 1, \ldots)$; and

$$\mathsf{nat}^3 = (1^3, 2^3, 3^3, \ldots) = \mathsf{nat} \odot \mathsf{nat} \odot \mathsf{nat}$$

with

$$(\sigma \odot \tau)(0) = \sigma(0) \cdot \tau(0) \qquad (\sigma \odot \tau)' = \sigma' \odot \tau'$$

Formalising Moessner's theorem

$$\mathrm{nat}^3 = \Sigma \circ \mathit{Drop}_2 \circ \Sigma \circ \mathit{Drop}_3(\,\mathrm{nat}\,)$$

and where

$$\begin{aligned}
\Sigma(\sigma) &= (\sigma(0),\ \sigma(0)+\sigma(1),\ \sigma(0)+\sigma(1)+\sigma(2),\ldots) \\
\mathit{Drop}_2(\sigma) &= (\sigma(0),\sigma(2),\sigma(4),\ldots) \\
\mathit{Drop}_3(\sigma) &= (\sigma(0),\sigma(1),\ \sigma(3),\sigma(4),\ \sigma(6),\sigma(7),\ \ldots)
\end{aligned}$$

can all be specified by elementary stream diff. equations.

Formalising Moessner's theorem

$$\text{nat}^3 = \Sigma \circ \mathit{Drop}_2 \circ \Sigma \circ \mathit{Drop}_3(\,\text{nat}\,)$$

and where

$$\begin{aligned}
\Sigma(\sigma) &= (\sigma(0),\ \sigma(0)+\sigma(1),\ \sigma(0)+\sigma(1)+\sigma(2),\ldots) \\
\mathit{Drop}_2(\sigma) &= (\sigma(0),\sigma(2),\sigma(4),\ldots) \\
\mathit{Drop}_3(\sigma) &= (\sigma(0),\sigma(1),\ \sigma(3),\sigma(4),\ \sigma(6),\sigma(7),\ \ldots)
\end{aligned}$$

can all be specified by elementary stream diff. equations.

Proving Moessner's theorem

$$\mathrm{nat}^3 = \Sigma \circ \mathit{Drop}_2 \circ \Sigma \circ \mathit{Drop}_3(\,\mathrm{nat}\,)$$

It now suffices to construct a bisimulation R with

$$\langle \mathrm{nat}^3,\ \Sigma \circ \mathit{Drop}_2 \circ \Sigma \circ \mathit{Drop}_3(\,\mathrm{nat}\,)\rangle \in R$$

Easy, using the previous stream differential equations . . .

Proving Moessner's theorem

$$\mathrm{nat}^3 = \Sigma \circ \mathit{Drop}_2 \circ \Sigma \circ \mathit{Drop}_3(\,\mathrm{nat}\,)$$

It now suffices to construct a bisimulation R with

$$\langle \mathrm{nat}^3,\ \Sigma \circ \mathit{Drop}_2 \circ \Sigma \circ \mathit{Drop}_3(\,\mathrm{nat}\,)\rangle \ \in R$$

Easy, using the previous stream differential equations . . .

Proving Moessner's theorem

$$\mathrm{nat}^3 = \Sigma \circ \mathit{Drop}_2 \circ \Sigma \circ \mathit{Drop}_3(\,\mathrm{nat}\,)$$

It now suffices to construct a bisimulation R with

$$\langle \mathrm{nat}^3,\ \Sigma \circ \mathit{Drop}_2 \circ \Sigma \circ \mathit{Drop}_3(\,\mathrm{nat}\,)\rangle \in R$$

Easy, using the previous stream differential equations . . .

Proving Moessner's theorem

$$\mathrm{nat}^3 = \Sigma \circ \mathit{Drop}_2 \circ \Sigma \circ \mathit{Drop}_3(\,\mathrm{nat}\,)$$

Proof: We define R as the smallest set such that

(i) $\langle \mathrm{nat}^3,\ \Sigma \circ \mathit{Drop}_2 \circ \Sigma \circ \mathit{Drop}_3(\,\mathrm{nat}\,)\rangle \ \in R$

(ii) $\langle\, \mathrm{nat} \odot (\mathrm{nat} + \mathrm{ones})^2\,,\ \Sigma \circ \mathit{Drop}_2^0 \circ \Sigma \circ \mathit{Drop}_3^1(\,\mathrm{nat}\,)\,\rangle \ \in R$

(iii) if $\langle \sigma_1, \sigma_2\rangle \in R$ and $\langle \tau_1, \tau_2\rangle \in R$ then $\langle \sigma_1 + \tau_1,\ \sigma_2 + \tau_2\rangle \ \in R$

(iv) $\langle \sigma, \sigma\rangle \in R$ (all σ)

Then: R is a bisimulation relation.

Proving Moessner's theorem

$$\mathrm{nat}^3 = \Sigma \circ \mathit{Drop}_2 \circ \Sigma \circ \mathit{Drop}_3(\,\mathrm{nat}\,)$$

Proof: We define R as the smallest set such that

(i) $\langle \mathrm{nat}^3,\ \Sigma \circ \mathit{Drop}_2 \circ \Sigma \circ \mathit{Drop}_3(\,\mathrm{nat}\,)\rangle \in R$

(ii) $\langle\, \mathrm{nat} \odot (\mathrm{nat} + \mathrm{ones})^2\,,\ \Sigma \circ \mathit{Drop}_2^0 \circ \Sigma \circ \mathit{Drop}_3^1(\,\mathrm{nat}\,)\,\rangle \in R$

(iii) if $\langle \sigma_1, \sigma_2 \rangle \in R$ and $\langle \tau_1, \tau_2 \rangle \in R$ then $\langle \sigma_1 + \tau_1,\ \sigma_2 + \tau_2 \rangle \in R$

(iv) $\langle \sigma, \sigma \rangle \in R$ (all σ)

Then: R is a bisimulation relation.

Proving Moessner's theorem

$$\mathrm{nat}^3 = \Sigma \circ \mathit{Drop}_2 \circ \Sigma \circ \mathit{Drop}_3(\,\mathrm{nat}\,)$$

Proof: We define R as the smallest set such that

(i) $\langle \mathrm{nat}^3,\ \Sigma \circ \mathit{Drop}_2 \circ \Sigma \circ \mathit{Drop}_3(\,\mathrm{nat}\,)\rangle \in R$

(ii) $\langle\, \mathrm{nat} \odot (\mathrm{nat} + \mathrm{ones})^2\,,\ \Sigma \circ \mathit{Drop}_2^0 \circ \Sigma \circ \mathit{Drop}_3^1(\,\mathrm{nat}\,)\,\rangle \in R$

(iii) if $\langle \sigma_1, \sigma_2 \rangle \in R$ and $\langle \tau_1, \tau_2 \rangle \in R$ then $\langle \sigma_1 + \tau_1,\ \sigma_2 + \tau_2 \rangle \in R$

(iv) $\langle \sigma, \sigma \rangle \in R$ (all σ)

Then: R is a bisimulation relation.

Proving Moessner's theorem

$$\mathsf{nat}^3 = \Sigma \circ \mathit{Drop}_2 \circ \Sigma \circ \mathit{Drop}_3(\,\mathsf{nat}\,)$$

Proof: We define R as the smallest set such that

(i) $\langle \mathsf{nat}^3,\ \Sigma \circ \mathit{Drop}_2 \circ \Sigma \circ \mathit{Drop}_3(\,\mathsf{nat}\,)\rangle \in R$

(ii) $\langle\, \mathsf{nat} \odot (\mathsf{nat} + \mathsf{ones})^2\,,\ \Sigma \circ \mathit{Drop}_2^0 \circ \Sigma \circ \mathit{Drop}_3^1(\,\mathsf{nat}\,)\,\rangle \in R$

(iii) if $\langle \sigma_1, \sigma_2 \rangle \in R$ and $\langle \tau_1, \tau_2 \rangle \in R$ then $\langle \sigma_1 + \tau_1,\ \sigma_2 + \tau_2 \rangle \in R$

(iv) $\langle \sigma, \sigma \rangle \in R$ (all σ)

Then: R is a bisimulation relation.

Proving Moessner's theorem

$$\mathrm{nat}^3 = \Sigma \circ \mathit{Drop}_2 \circ \Sigma \circ \mathit{Drop}_3(\mathrm{nat})$$

Proof: We define R as the smallest set such that

(i) $\langle \mathrm{nat}^3,\ \Sigma \circ \mathit{Drop}_2 \circ \Sigma \circ \mathit{Drop}_3(\mathrm{nat})\rangle \in R$

(ii) $\langle\, \mathrm{nat} \odot (\mathrm{nat} + \mathrm{ones})^2\,,\ \Sigma \circ \mathit{Drop}_2^0 \circ \Sigma \circ \mathit{Drop}_3^1(\mathrm{nat})\,\rangle \in R$

(iii) if $\langle \sigma_1, \sigma_2\rangle \in R$ and $\langle \tau_1, \tau_2\rangle \in R$ then $\langle \sigma_1 + \tau_1,\ \sigma_2 + \tau_2\rangle \in R$

(iv) $\langle \sigma, \sigma\rangle \in R$ (all σ)

Then: R is a bisimulation relation.

Proving Moessner's theorem

The proof for all k: make one **big bisimulation**

Proof has been verified in theorem prover (COQ),
by Krebbers, Parlant, Silva.

Proving Moessner's theorem

The proof for all k: make one **big bisimulation**

Proof has been verified in theorem prover (COQ), by Krebbers, Parlant, Silva.

5. Discussion

- We take streams σ as **basic entities**, instead of focussing on their individual **elements** $\sigma(n)$.
- This prevents lots of unnecessary bookkeeping (cf. binomial coefficients).
- The (final) coalgebra structure of the set of streams has a natural interpretation in terms of a **calculus**, in analogy to classical calculus.
- There is initial evidence that this leads to efficient proofs that can be easily automated.

5. Discussion

- We take streams σ as **basic entities**, instead of focussing on their individual **elements** $\sigma(n)$.
- This prevents lots of unnecessary bookkeeping (cf. binomial coefficients).
- The (final) coalgebra structure of the set of streams has a natural interpretation in terms of a **calculus**, in analogy to classical calculus.
- There is initial evidence that this leads to efficient proofs that can be easily automated.

5. Discussion

- We take streams σ as **basic entities**, instead of focussing on their individual **elements** $\sigma(n)$.
- This prevents lots of unnecessary bookkeeping (cf. binomial coefficients).
- The (final) coalgebra structure of the set of streams has a natural interpretation in terms of a **calculus**, in analogy to classical calculus.
- There is initial evidence that this leads to efficient proofs that can be easily automated.

5. Discussion

- We take streams σ as **basic entities**, instead of focussing on their individual **elements** $\sigma(n)$.
- This prevents lots of unnecessary bookkeeping (cf. binomial coefficients).
- The (final) coalgebra structure of the set of streams has a natural interpretation in terms of a **calculus**, in analogy to classical calculus.
- There is initial evidence that this leads to efficient proofs that can be easily automated.

5. Discussion

- We take streams σ as **basic entities**, instead of focussing on their individual **elements** $\sigma(n)$.
- This prevents lots of unnecessary bookkeeping (cf. binomial coefficients).
- The (final) coalgebra structure of the set of streams has a natural interpretation in terms of a **calculus**, in analogy to classical calculus.
- There is initial evidence that this leads to efficient proofs that can be easily automated.