

# MONOIDAL STREAMS FOR DATAFLOW PROGRAMMING

---

ELENA DI LAVORE, GIOVANNI DE FELICE, MARIO ROMÁN

TALLINN UNIVERSITY OF TECHNOLOGY / U. OF OXFORD

Supported by the EU Estonian IT Academy.



JUNE 20<sup>st</sup>, 2022 LiCS22

# MONOIDAL STREAMS FOR DATAFLOW PROGRAMMING



Elena Di Lavoro

Tallinn University  
of Technology.



Giovanni de Felice

University of Oxford +  
Quantinuum



Mario Román

Tallinn University  
of Technology.



# PART 0: DATAFLOW PROGRAMMING

---

# MOTIVATION: DATAFLOW PROGRAMMING

Dataflow programming is a paradigm for repeated processes: every declaration is a sequence of values.

$$\begin{aligned} \text{fib} &= 0 \text{ FBY } (1 \text{ FBY WAIT fib} + \text{fib}) \\ \text{nat} &= 0 \text{ FBY } (1 + \text{nat}) \end{aligned}$$

- Elegant recursive dataflow syntax.
- Signal flow, 'trace-like' diagrams.

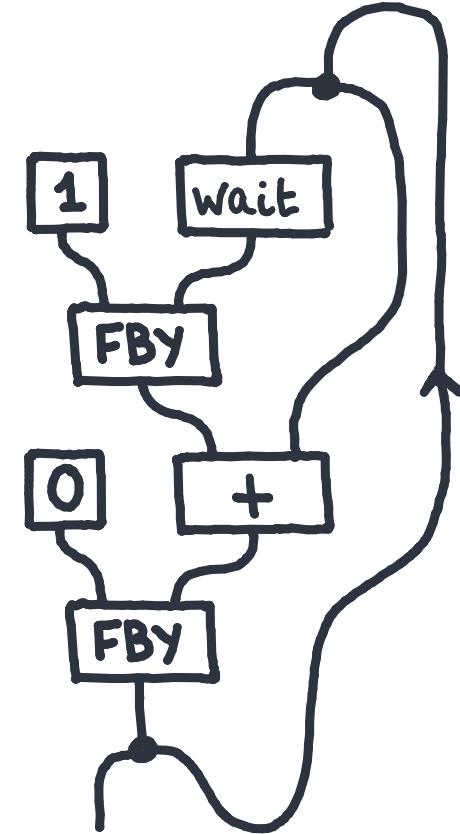


FIG. Signal flow graph.

E.g. LUSTRE, LUCID.  Ashcroft, Wadge, 85

# MOTIVATION: DATAFLOW PROGRAMMING

Dataflow programming is a paradigm for repeated processes: every declaration is a sequence of values.

{"followed by"}

$\text{fib} = 0 \text{ FBY } (1 \text{ FBY WAIT fib} + \text{fib})$

$\text{nat} = 0 \text{ FBY } (1 + \text{nat})$

"delayed stream"

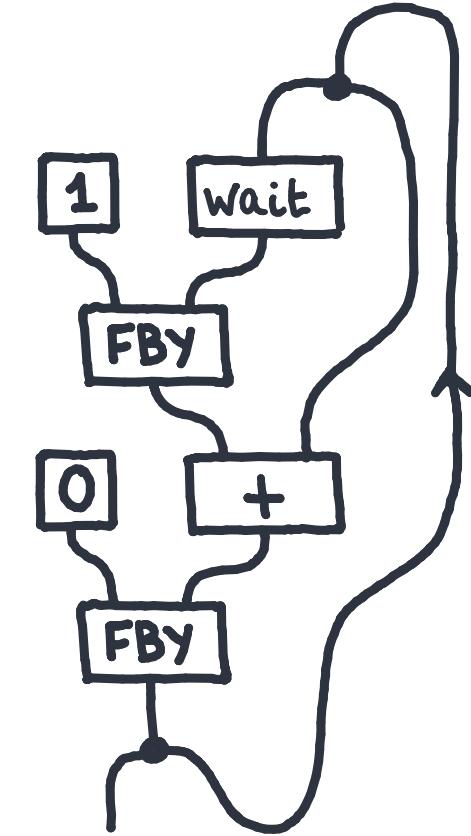


FIG. Signal flow graph.

E.g. LUSTRE, LUCID.  Ashcroft, Wadge, 85

# SEMANTICS: STREAM FUNCTIONS

---

DEFINITION. A *causal stream function*  $f: \mathbb{X} \rightarrow \mathbb{Y}$  is a family of functions, indexed by natural number time, from all past inputs to the current output.

$$\begin{aligned}f_0 &: X_0 \rightarrow Y_0 \\f_1 &: X_0 \times X_1 \rightarrow Y_1 \\f_2 &: X_0 \times X_1 \times X_2 \rightarrow Y_2 \\&\dots\end{aligned}$$

## PROPERTIES:

- a *delay functor* taking  $\mathbb{X} = (X_0, X_1, X_2, \dots)$  into  $\partial\mathbb{X} = (1, X_0, X_1, \dots)$ ;
- a *feedback-like operator* taking  $\partial S \otimes \mathbb{X} \rightarrow S \otimes \mathbb{Y}$  into  $\mathbb{X} \rightarrow \mathbb{Y}$ ;
- *coalgebraic reasoning* and *coinductive arguments*.

 Sprunger, Katsumata 19

CATEGORICALLY: The cokleisli category of the non-empty list monoidal comonad,

$$\text{List}^+: [\mathbf{N}, \mathbf{SET}] \rightarrow [\mathbf{N}, \mathbf{SET}], \quad \text{List}^+(\mathbb{X})_n = \prod_{i=0}^n X_i.$$

 Uustalu, Vene 08

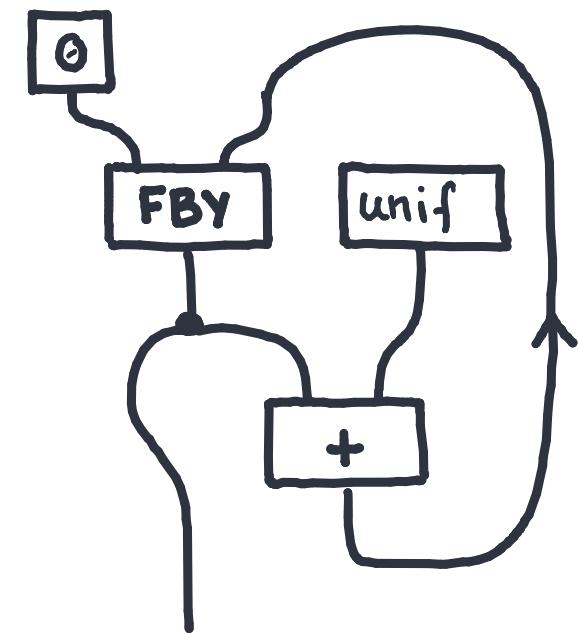
# OTHER THEORIES OF PROCESSES

What about other theories of processes?

$$\text{walk} = \emptyset \text{ FBY UNIFORM}\{-1,1\} + \text{walk}$$

E.g. stochastic functions  $\leadsto$  stochastic streams.

Given a theory of processes,  
we construct a theory of streams  
over them.



# OTHER THEORIES OF PROCESSES

What about other theories of processes?

$$X = (0 \text{ FBy } X) + (1 \text{ FBy } Y)$$

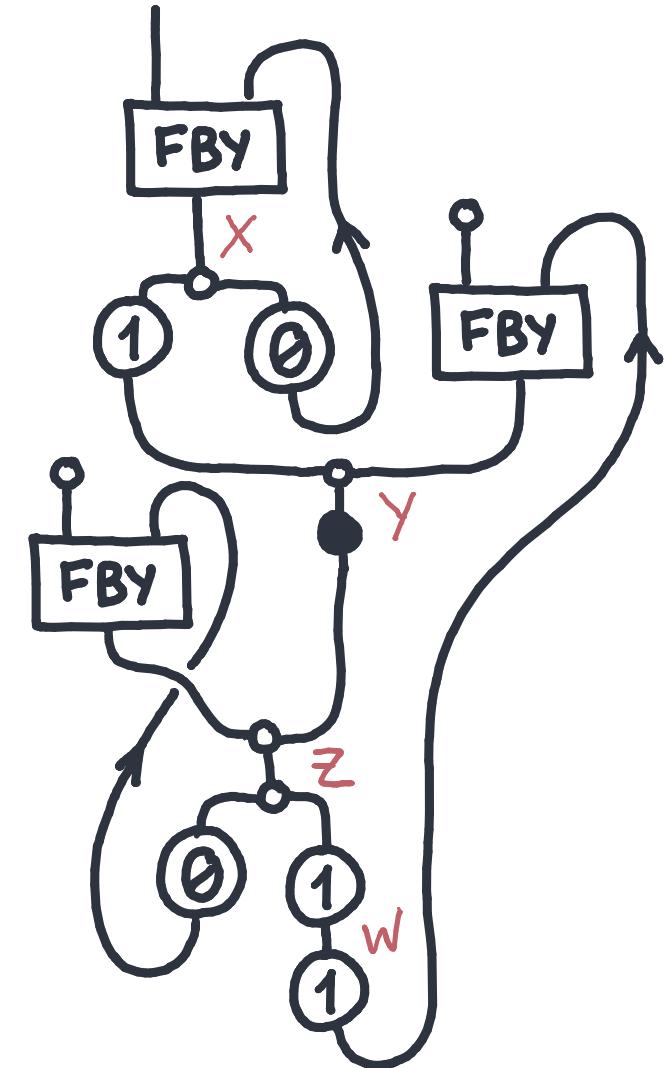
$$Y = \bullet(Z)$$

$$Z = (0 \text{ FBy } Z) + (1 \text{ FBy } W)$$

$$W = (1 \text{ FBy } Y)$$

E.g. non-deterministic functions  $\rightsquigarrow$  non-deterministic streams.

Given a theory of processes,  
we construct a theory of streams  
over them.



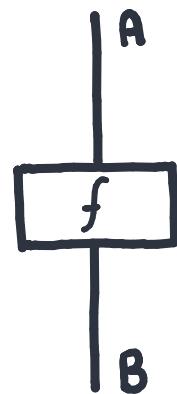
# PART 1: MONOIDAL CATEGORIES

---

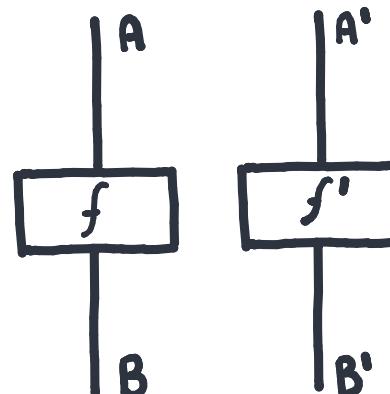
# MONOIDAL CATEGORIES: PROCESS THEORIES

---

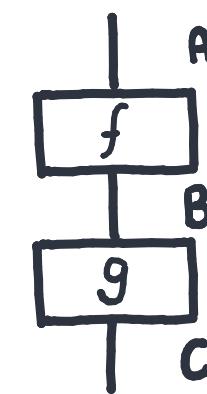
Monoidal categories are an algebra of parallel and sequential composition.  
String diagrams are the internal language of monoidal categories.



Process



Parallel composition



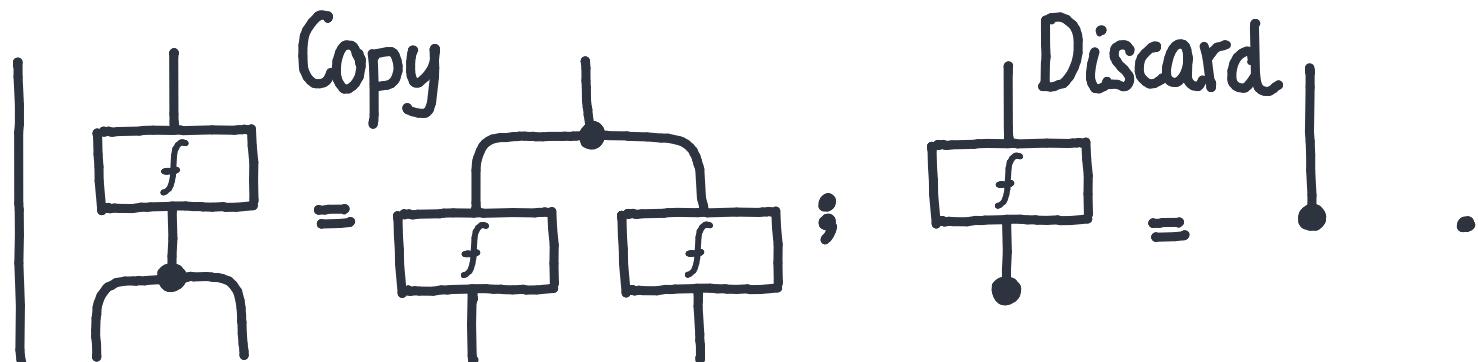
Sequential composition



Bénabou

# MONOIDAL CATEGORIES: CARTESIAN

DEFINITION. Monoidal categories are cartesian when processes can be copied and discarded uniformly.



Deterministic functions  
Stochastic functions  
Partial functions  
Relations

✓  
✗  
✓  
✗

✓  
✓  
✗  
✗



T. Fox

# MONOIDAL DATAFLOW PROGRAMMING

---

THEOREM (DLdFR). The non-empty list functor  $\text{List}^+(\mathbb{X})_n = \bigotimes_{i=0}^n \mathbb{X}_i$  is a monoidal comonad if and only if  $(\otimes)$  is a cartesian product.

What about other theories of processes?



What about other, non-cartesian, monoidal categories?

# SYNOPSIS

Today, given any symmetric monoidal category  $(\mathcal{C}, \otimes, I)$ , we will build a symmetric monoidal category of stream processes  $\text{Stream}(\mathcal{C})$  such that

- $\text{Stream}(\mathcal{C})$  has an id-on objs functor from  $[N, \mathcal{C}]$ ;
- $\text{Stream}(\mathcal{C})$  has a delay monoidal functor,  $\mathcal{D}$ ;
- $\text{Stream}(\mathcal{C})$  has delayed feedback taking  $\mathcal{D}S \otimes X \rightarrow S \otimes Y$  into  $X \rightarrow Y$ ;
- $\text{Stream}(\mathcal{C})$  has a coalgebraic description;
- $\text{Stream}(\mathcal{C})$  is cartesian when  $\mathcal{C}$  is;
- $\text{Stream}(SET)$  is the classical causal streams;
- $\text{Stream}(STOCH)$  is causal discrete stochastic processes.
- $\text{Stream}(\mathcal{C})$  is symm. premonoidal, effectful or Freyd when  $\mathcal{C}$  is.

Three definitions in terms of universal properties, and three constructions.

# SYNOPSIS

---

Three definitions from universal properties, and three explicit constructions.  
Each one a quotient of the previous.

1. Intensional streams, a first naïve version. Fail to form a category.
2. Extensional streams, a free category with feedback.
3. Observational streams, definitive solution to a fixpoint equation.

Two known particular cases, and an avenue for more.

1. Cartesian monoidal streams  $(\text{Set}, \times)$  are causal functions  
(as in Uustalu-Vene, Sprunger-Jacobs).
2. Stochastic streams  $(\text{Kl}(\mathbb{D}), \times)$  are controlled stochastic processes  
(classical in the literature).
3. Kleisli streams of strong monads. Premonoidal categories in general.

(Intensional)

## PART 2: MONOIDAL STREAMS

---

# STREAMS AND STREAM FUNCTIONS

"A **stream** of types  $A = (A_0, A_1, A_2, \dots)$  is an element of  $A_0$  together with a **stream** of types  $A^+ = (A_1, A_2, A_3, \dots)$ ."

$$S(A) \cong A_0 \times S(A^+).$$

# STREAMS AND STREAM FUNCTIONS

---

"A **stream** of types  $A = (A_0, A_1, A_2, \dots)$  is an element of  $A_0$  together with a **stream** of types  $A^+ = (A_1, A_2, A_3, \dots)$ ."

$$S(A) \cong A_0 \times S(A^+).$$

By **Adámek's Theorem**, the candidate solution is

$$\lim_{n \in \mathbb{N}} (1 \leftarrow A_0 \leftarrow A_0 \times A_1 \leftarrow A_0 \times A_1 \times A_2 \leftarrow \dots) = \prod_{n=0}^{\infty} A_n;$$

and it is a solution,  $A_0 \times \prod_{n=1}^{\infty} A_n \cong \prod_{n=0}^{\infty} A_n$ .



# STREAMS AND STREAM FUNCTIONS

---

“A stream function from  $\mathbb{X} = (X_0, X_1, X_2, \dots)$  to  $\mathbb{Y} = (Y_0, Y_1, Y_2, \dots)$  is a function  $X_0 \rightarrow Y_0$  communicating along a memory channel  $M$  with a stream function from  $\mathbb{X}^+ = (X_1, X_2, X_3, \dots)$  to  $\mathbb{Y}^+ = (Y_1, Y_2, Y_3, \dots)$ .”

$$T(\mathbb{X}; \mathbb{Y}) = \sum_{M \in \text{SET}} \text{hom}(X_0, M \times Y_0) \times T(M \times X_1, X_2, X_3, \dots; Y_1, Y_2, Y_3, \dots).$$

By Adamek's Theorem, the candidate solution is

$$\lim_{n \in \mathbb{N}} (1 \leftarrow \text{hom}(X_0, Y_0) \leftarrow \text{hom}(X_0, Y_0) \times \text{hom}(X_0 \times X_1, Y_1) \leftarrow \dots) = \prod_{n=0}^{\infty} \text{hom}(X_0 \times \dots \times X_n, Y_n);$$



# MONOIDAL STREAMS

"A monoidal stream from  $\mathbb{X} = (X_0, X_1, X_2, \dots)$  to  $\mathbb{Y} = (Y_0, Y_1, Y_2, \dots)$  is a morphism  $X_0 \rightarrow Y_0$  communicating along a memory channel  $M$  with a monoidal stream from  $\mathbb{X}^+ = (X_1, X_2, X_3, \dots)$  to  $\mathbb{Y}^+ = (Y_1, Y_2, Y_3, \dots)$ ."

$$T(\mathbb{X}; \mathbb{Y}) = \sum_{M \in SET} \text{hom}(X_0, M \otimes Y_0) \times T(M \otimes X_1, X_2, X_3, \dots; Y_1, Y_2, Y_3, \dots).$$

By Adamek's Theorem, the candidate solution is

$$\lim_{n \in \mathbb{N}} (1 \leftarrow \sum_n \sum_{M_0} \text{hom}(X_0, M_0 \otimes Y_0) \leftarrow \sum_{M_0, M_1} \text{hom}(X_0, M_0 \otimes Y_0) \times \text{hom}(M_0 \otimes X_1, M_1 \otimes Y_1) \leftarrow \dots) = \\ \sum_{M: [N, C]} \prod_{n \in \mathbb{N}} \text{hom}(M_{n-1} \otimes X_n, M_n \otimes Y_n).$$

# MONOIDAL STREAMS

---

DEFINITION (DLdFR). An (intensional) **monoidal stream**  $\mathbb{X} \rightarrow \mathbb{Y}$  is a family of objects  $M_0, M_1, M_2, \dots$  and a family of morphisms  $f_n : M_{n-1} \otimes X_n \rightarrow M_n \otimes Y_n$ .

$$T(\mathbb{X}, \mathbb{Y}) = \sum_{M: [N, C]} \prod_{n \in N} \text{hom}(M_{n-1} \otimes X_n, M_n \otimes Y_n).$$

THEOREM (DLdFR). The set of monoidal streams, depending on inputs and outputs, is the terminal fixpoint of

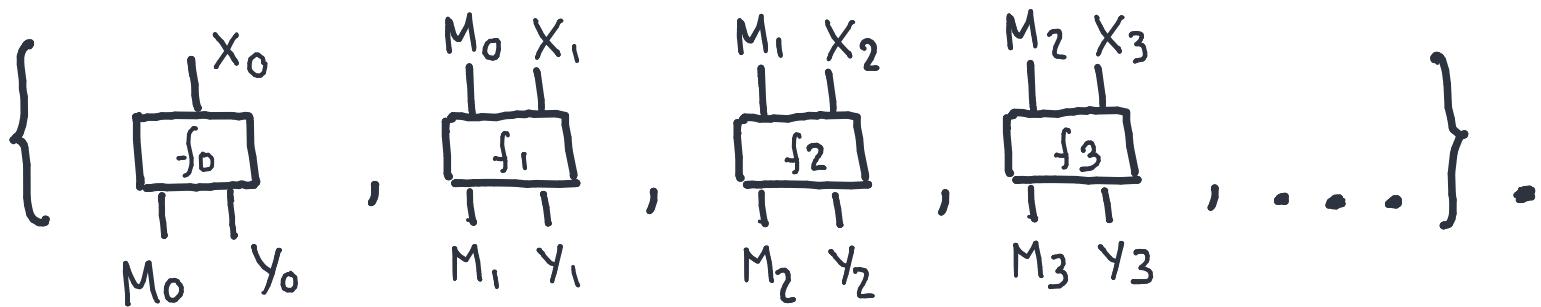
$$T(\mathbb{X}, \mathbb{Y}) = \sum_{M \in \text{SET}} \text{hom}(X_0, M \otimes Y_0) \times T(M \otimes X_1, X_2, X_3, \dots; Y_1, Y_2, Y_3, \dots).$$

# MONOIDAL STREAMS

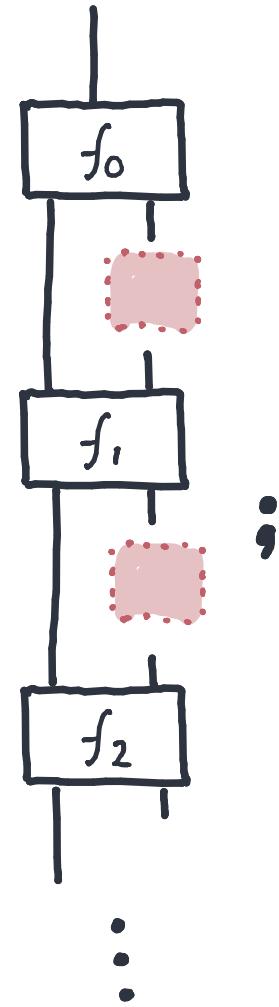
How to interpret a monoidal stream? In diagrams,

$$(f_n: M_{n-1} \otimes X_n \rightarrow M_n \otimes Y_n)$$

is

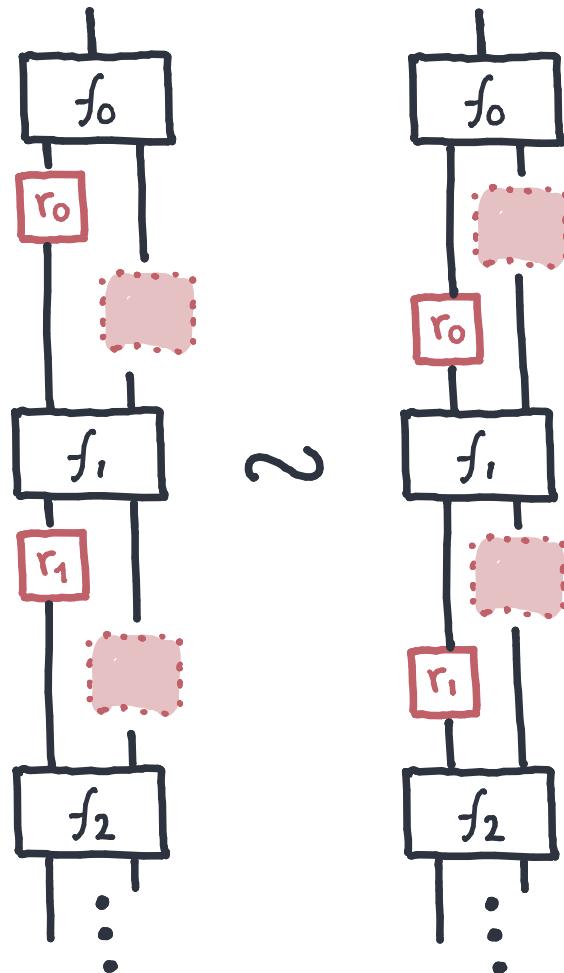


and yields the following "open diagram".



# DINATURALITY

Monoidal streams are too explicit.



Dinaturality generalizes categorical naturality: it is used to compose profunctors, so it appears implicitly when communicating processes in dataflow programming.  
Hildebrandt, Panangaden, Winskel. 97.

DEFINITION. An (extensional) monoidal stream is an equivalence class of intensional streams under the minimal equivalence relation containing  $(\sim)$ .

$$\sum_{M:[N,C]} \prod_{n \in N} \text{hom}(M_{n-1} \otimes X_n, M_n \otimes Y_n) / \langle \sim \rangle$$
$$= \int^{M:[N,C]} \prod_{n \in N} \text{hom}(M_{n-1} \otimes X_n, M_n \otimes Y_n)$$

(Coinductive)

# PART 3 : MONOIDAL STREAMS

---

# COINDUCTIVE MONOIDAL STREAMS

---

THEOREM (DLdFR). Monoidal streams (with observational eq.) are the final coalgebra of

$$Q(X, Y) \cong \int^{M:C} \text{hom}(X_0, M \otimes Y_0) \times Q(M \otimes X_1, X_2, X_3, \dots; Y_1, Y_2, Y_3, \dots).$$

DEFINITION (DLdFR). A monoidal stream  $f \in \text{Stream}(X_0, X_1, \dots; Y_0, Y_1, \dots)$  is

- a memory  $M(f) \in C$
- a  $\text{now}(f) : X_0 \rightarrow M(f) \otimes Y_0$ ,
- and a  $\text{later}(f) \in \text{Stream}(M \otimes X_1, X_2, \dots; Y_1, Y_2, \dots)$ .

Quotiented by  $f \approx g$ , meaning

- the existence of  $r : M(f) \rightarrow M(g)$ ,

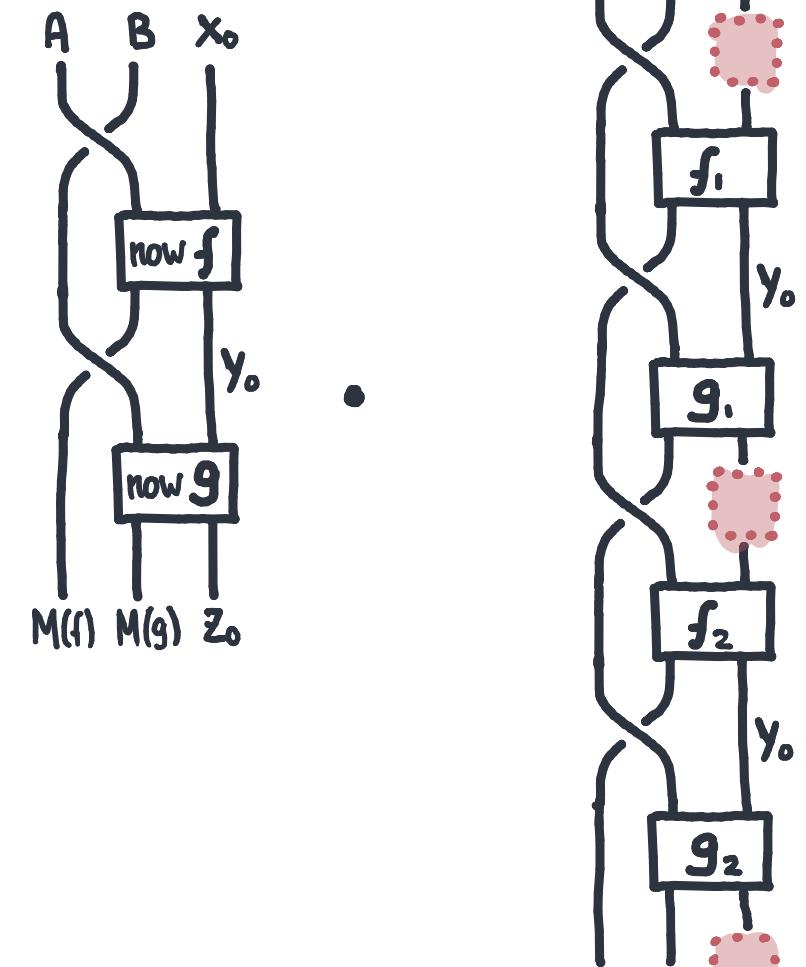
- such that  $\text{now}(f); (r \otimes \text{id}) = \text{now}(g)$ ,
- and such that  $\text{later}(f) \approx r \cdot \text{later}(g)$ .

# COINDUCTIVE MONOIDAL STREAMS

---

SEQUENTIAL COMPOSITION WITH MEMORY of  $f \in \text{Stream}(A \cdot X, Y)$  and  $g \in \text{Stream}(B \cdot Y, Z)$  is written as  $(f^A; g^B) \in \text{Stream}(A \otimes B \cdot X, Z)$ , and defined by

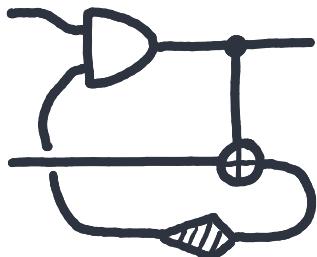
- $M(f^A; g^B) = M(f) \otimes M(g)$ ;
- $\text{later}(f^A; g^B) = \text{later}(f)^{M(f)} ; \text{later}(g)^{M(g)}$ , by coinduction;
- $\text{now}(f^A; g^B) =$



# FURTHER WORK

---

Coinduction via string diagrams.



Can we compare  
with the graphical  
language here?

Carette, de Visme, Perdrix.

Premonoidal categories.

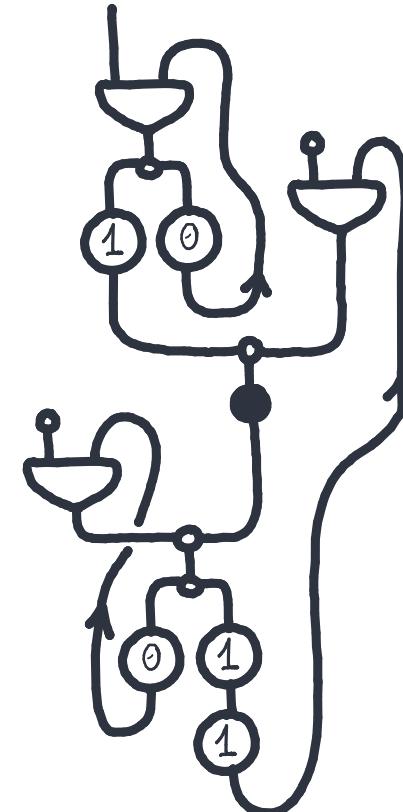
Restricting to central dinaturality.

Power, Robinson

Román. Open Diagrams.

Expressivity of monoidal streams.

Systems of equations for  
 $\omega$ -regular languages are  
monoidal streams over a  
suitable category of total  
relations.



$0^*1(0^*11)^\omega$

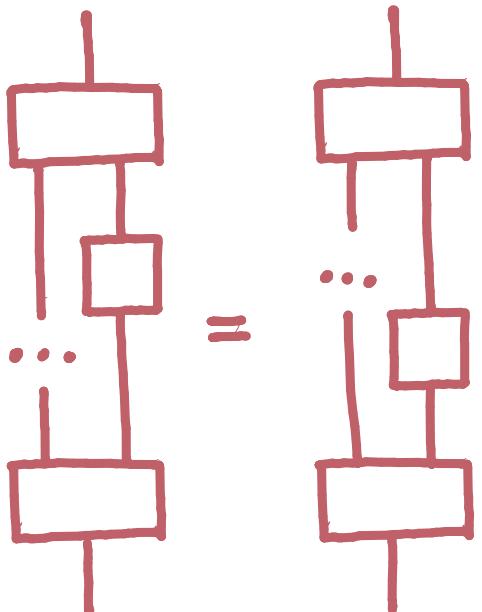
Park.

# CONCLUSIONS

---

TAKE HOME

- Given a theory of processes, we construct a theory of streams over them.  
 $(C, \otimes, I) \rightarrow (\text{Stream}_C, \otimes^N, I)$
- The key idea is to quotient by **dinaturality** before using coalgebra.



- We recover classical streams and stochastic processes.
- We can reason **coinductively**, as we do with streams.
- Implementation and a toy language are direct from the universal properties.

# MONOIDAL STREAMS FOR DATAFLOW PROGRAMMING



Elena Di Lavoro

Tallinn University  
of Technology.



Giovanni de Felice

University of Oxford +  
Quantinuum



Mario Román

Tallinn University  
of Technology.



END

# RELATED WORK

---



Katis-Sabadini-Walters. Categories with feedback, missing delay.



Sprunguer-Katsumata, Ghica-Kaye. Finite memory or cartesian streams.



Uustalu-Vene. Cartesian streams, distributive laws for effects.



Hughes-Paterson. ArrowLoop are similar, but for traces.



Carette-De Visme-Perdrix. Syntax for similar streams.



Many others. Categorical dataflow. Functional reactive programming.



Román. Open diagrams via Coend Calculus.

(Extensional)

## PART 3 : MONOIDAL STREAMS

---

I like **dinaturality**, but you may not; so let me justify it.

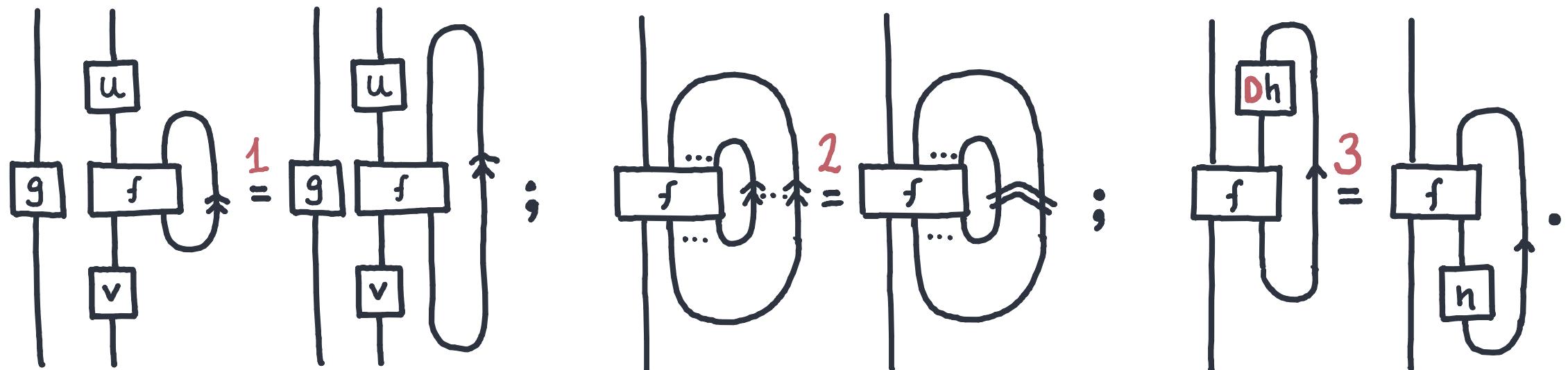
# FEEDBACK

---

Katis, Sabadini, Walters 97  
Hoshino, Muroya, Hasuo 14  
Bonchi, Sobociński, Zanasi 14

Feedback monoidal categories, axiomatize signal flow graphs using a guarded feedback operator  $\text{fbk}: \text{hom}(\text{DS} \otimes \text{A}, \text{S} \otimes \text{B}) \rightarrow \text{hom}(\text{A}, \text{B})$ .

AXIOMS.

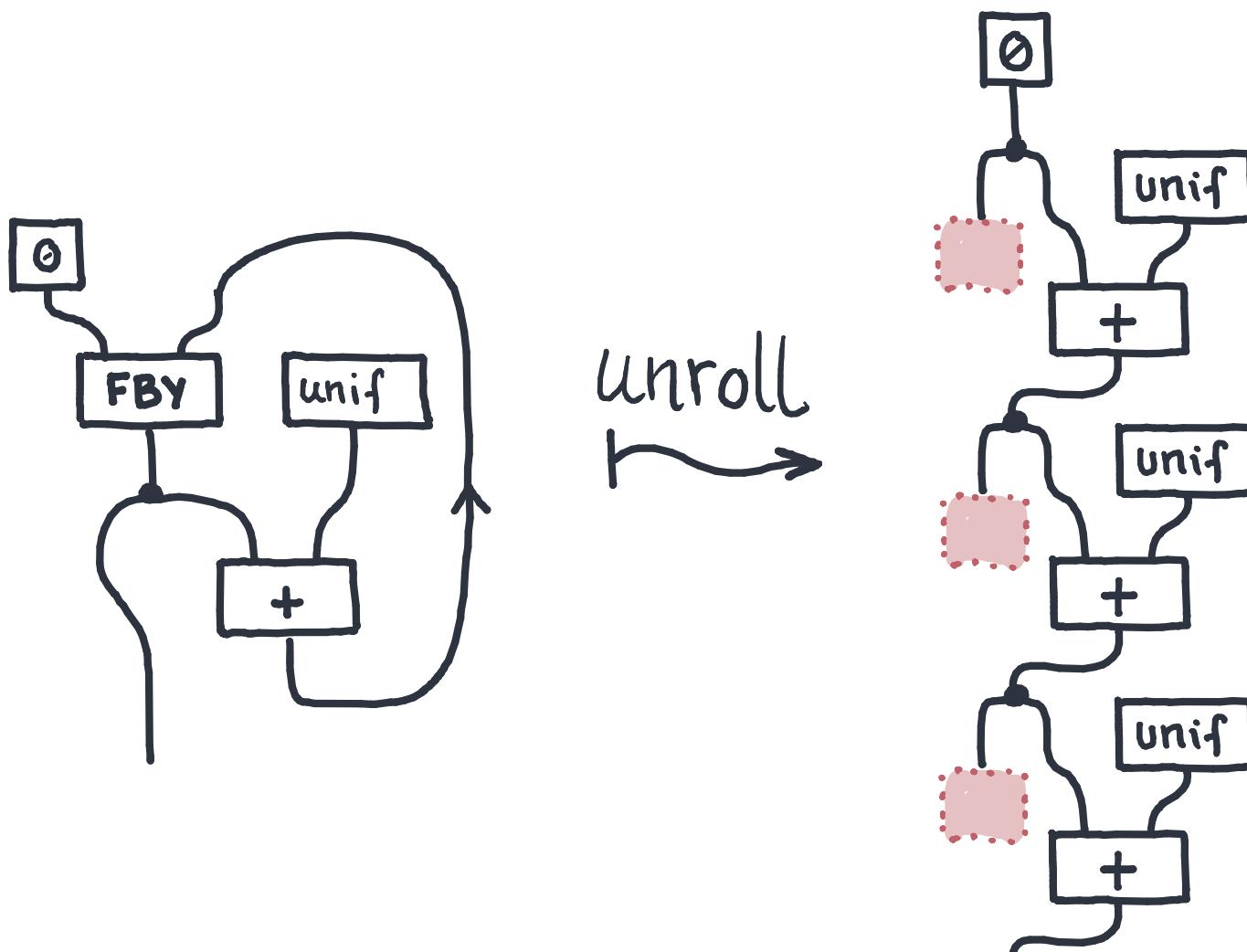


THEOREM (DLdFR). Extensional streams are the morphisms of the free category with feedback over  $\partial: [N, C] \rightarrow [N, C]$ ,  $\partial(A_0, A_1, A_2, \dots) = (I, A_0, A_1, \dots)$ .

# FEEDBACK

---

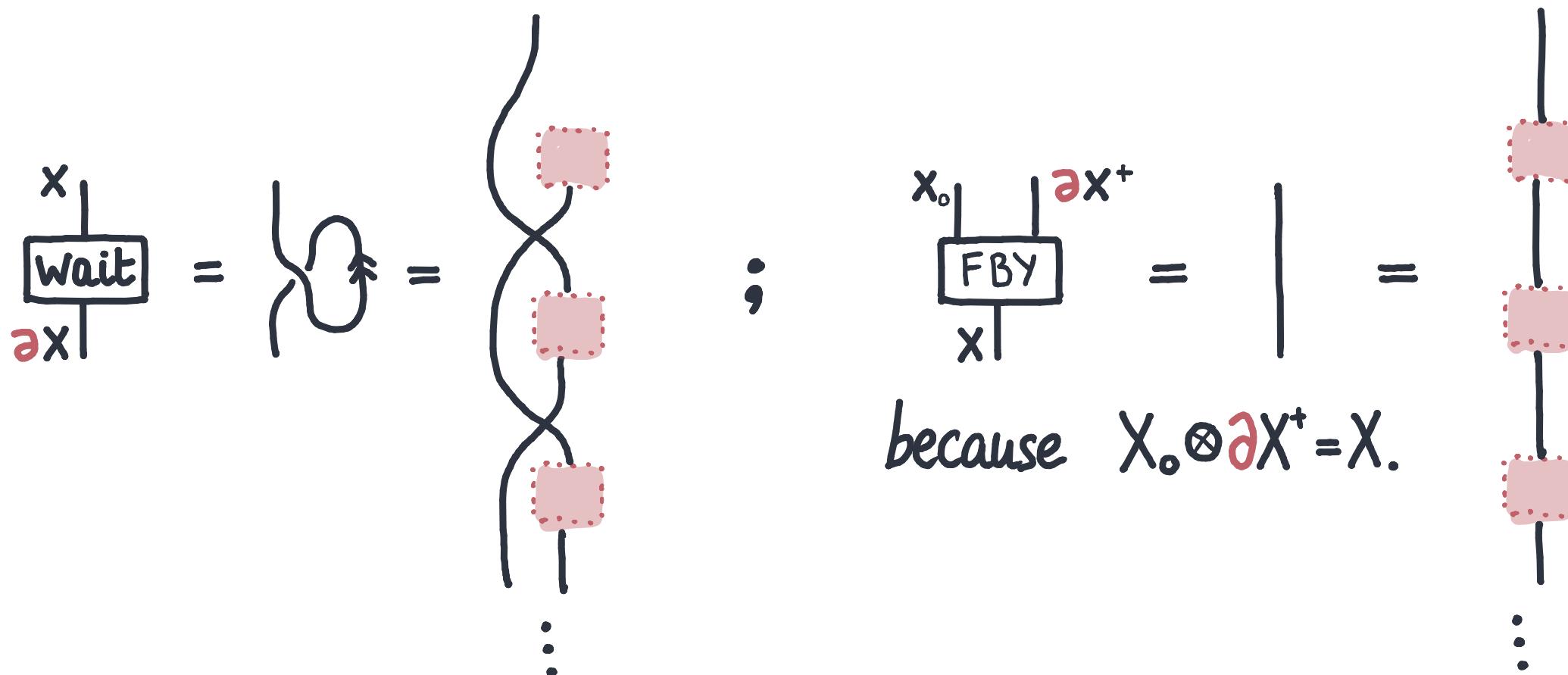
Syntax (free category with feedback) to semantics (monoidal streams).



# FEEDBACK

---

Dataflow syntax (**FBY**, **WAIT**) has meaning in terms of feedback and [IN,C].

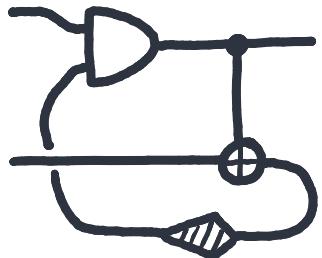


# FURTHER WORK

---



Carette, de Visme, Perdrix.



Can we compare  
with the graphical  
language here?

Delayed Trace.



Kaye, Sprunger, Ghica

Better treatment of the finite  
memory case.



Cruttwell, Gallagher, Lemay, Pronk  
Blute, Cockett, Lemay, Seely

Can we preserve the differential  
structure from cartesian to  
monoidal?



Spivak Myers Shapiro, Spivak  
Smithie Hornischer.

How much can we recover in terms  
of (continuous) dynamical systems?  
How much of the coalgebra in terms  
of polynomial functors?



Power, Robinson

Premonoidal streams are arguably  
more interesting.



Román. Open Diagrams.

# PART 5 : EXAMPLES

# STOCHASTIC PROCESSES

---

DEFINITION. A **controlled stochastic process**  $f: \mathbb{X} \rightarrow \mathbb{Y}$  is a family of stochastic functions  $f_n: X_0 \times \dots \times X_n \rightarrow D(Y_0 \times \dots \times Y_n)$  satisfying

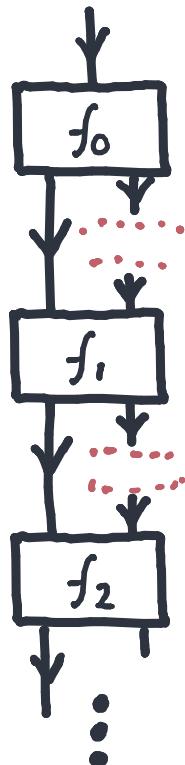
$$\begin{array}{ccc} X_0 \times \dots \times X_{n+1} & \xrightarrow{f_{n+1}} & D(Y_0 \times \dots \times Y_{n+1}) \\ \downarrow^n & & \downarrow D_n \\ X_0 \times \dots \times X_n & \xrightarrow{f_n} & D(Y_0 \times \dots \times Y_n). \end{array}$$

**causality:** the future  $X_{n+1}$  should not influence the past  $Y_0, \dots, Y_n$ .

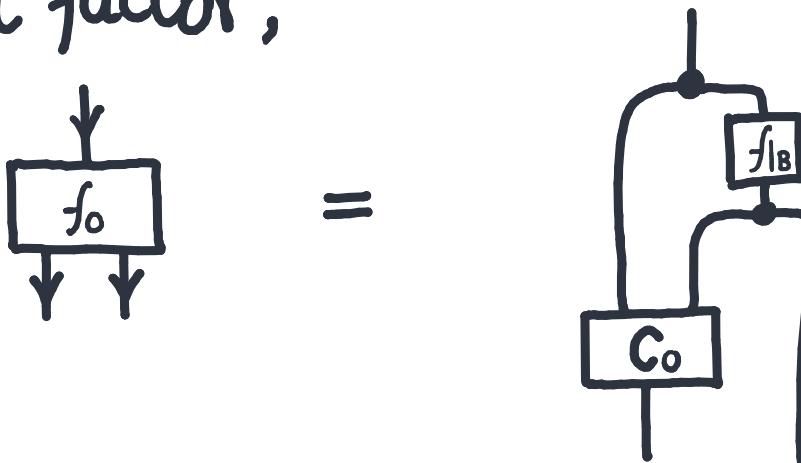
THEOREM (DLdFR). Monoidal streams over  $KL(D)$  coincide with controlled stochastic processes.

PROOF. Non trivial. Somehow, the causality condition means that the family can be written *uniquely* as a monoidal stream.

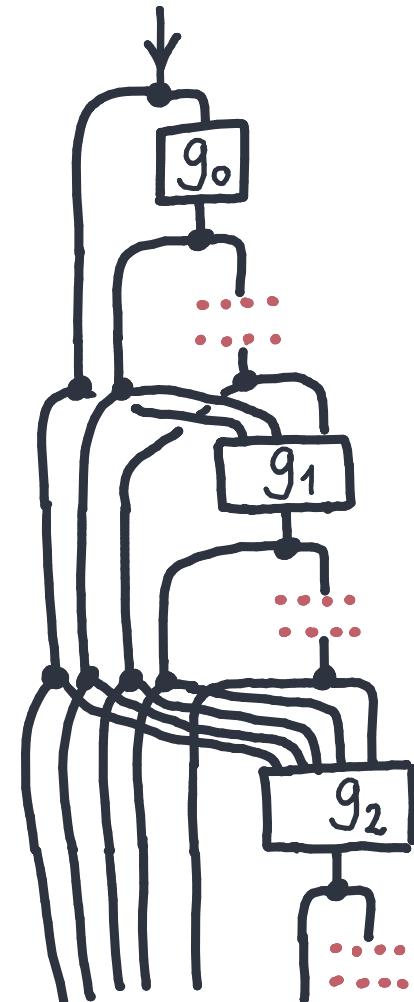
# STOCHASTIC MONOIDAL STREAMS



In **Markov categories** with conditionals,  
we can factor,

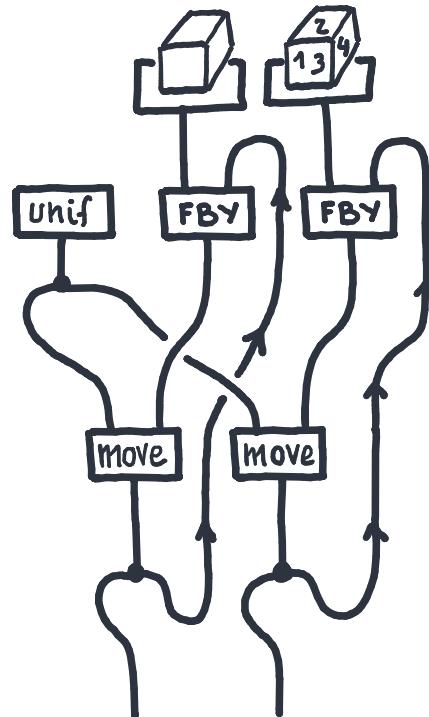
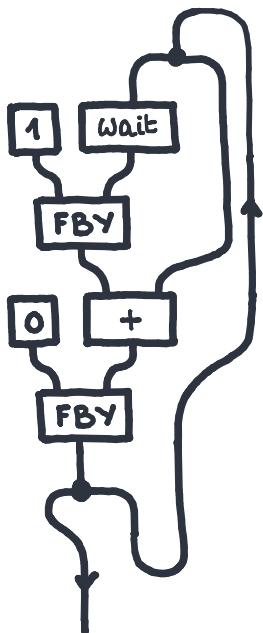
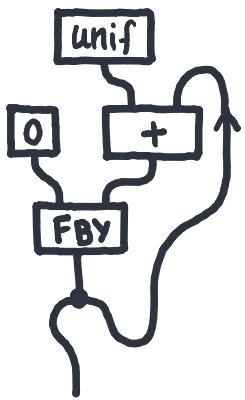


We use this to show that monoidal streams of stochastic functions are the same as **controlled stochastic processes**.



# EXTRA : IMPLEMENTATION

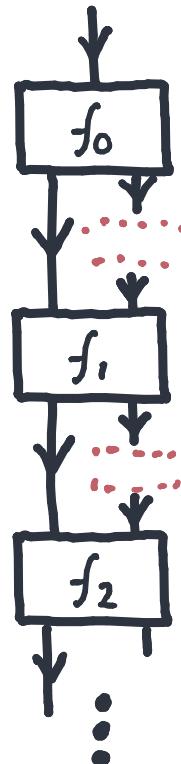
# EXAMPLES



# CARTESIAN MONOIDAL STREAMS

---

THEOREM. In cartesian monoidal categories, monoidal streams are causal stream functions.



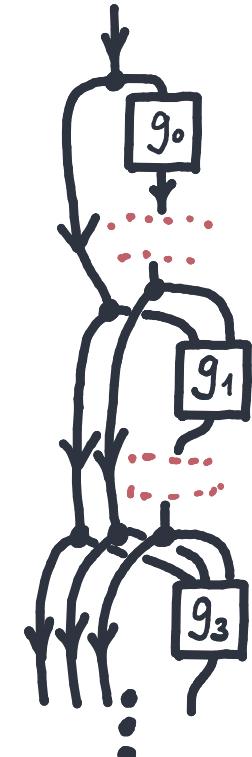
$S(X, Y)$

$$\cong \int^{M:C} \text{hom}(X_0, M \times Y_0) \times S(M \times X_1, X_2, X_3, \dots; Y_1, Y_2, Y_3, \dots)$$

$$\cong \int^{M:C} \text{hom}(X_0, M \times Y_0) \times S(M \times X_1, X_2, X_3, \dots; Y_1, Y_2, Y_3, \dots)$$

$$\cong \int^{M:C} \text{hom}(X_0, M) \times \text{hom}(X_0, Y_0) \times S(M \times X_1, X_2, X_3, \dots; Y_1, Y_2, Y_3, \dots)$$

$$\cong \text{hom}(X_0, Y_0) \times S(X_0 \times X_1, X_2, X_3, \dots; Y_1, Y_2, Y_3, \dots)$$

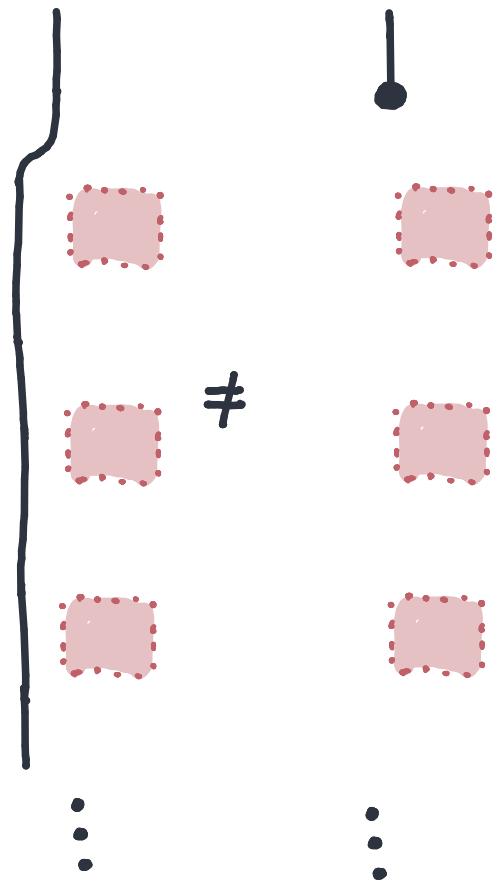
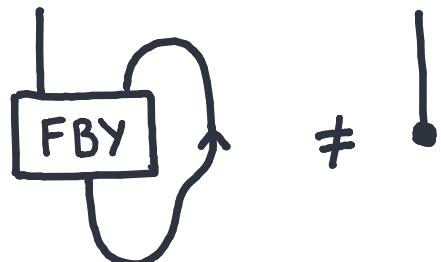


# OBSERVATIONAL EQUIVALENCE

---

So, we want, at least, extensional equivalence. Can we refine it?

Saving to memory without outputting is, observationally, the same as discarding. However, no amount of sliding will help us equating these two.

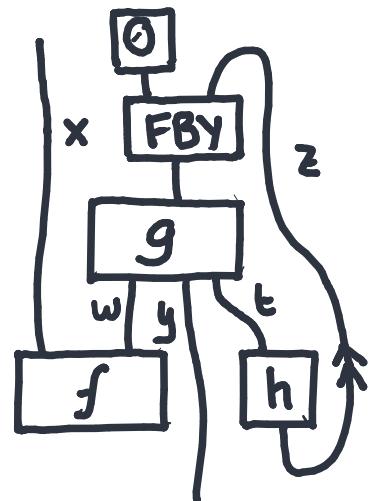


Two streams are observationally equal if their nth truncations can be made equal.

# IMPLEMENTATION

---

Type theory for sym. monoidal categories with feedback.



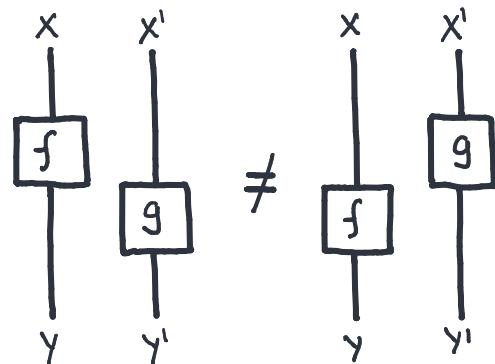
FBK z.  
SPLIT  $g(\theta \text{FBY } h(z)) \rightarrow [w, y, t]$  IN  
SPLIT  $f \times w \rightarrow []$  IN  
RETURN y.

Obvious candidate: Haskell **Arrows** give notation for Set-based Freyd categories.

- **ArrowLoop** is for traced categories, in theory; it works for feedback.
- Github: [mroman42/arrow-streams](#).
- **Agda** is a good candidate for coinduction.

# PREMONOIDAL CATEGORIES

A sym. premonoidal category  $(\mathcal{C}, \otimes, I)$  is a sym. monoidal category without the interchange law.



They usually have a family of "pure" morphisms that do satisfy interchange, forming a monoidal  $\mathbb{V}$ .

$$\mathbb{V} \rightarrow \mathcal{C}$$

PURE — id-on-objects functor → EFFECTFUL

This is called a Freyd category.

DEFINITION. The set of premonoidal streams in a Freyd category  $\mathbb{V} \rightarrow \mathcal{C}$  is the final fixpoint of

$$Q(X, Y) \cong \int^{M: \mathbb{V}} \text{hom}_{\mathcal{C}}(X_0, M \otimes Y_0) \times Q(M \otimes X_1, X_2, X_3, \dots; Y_1, Y_2, Y_3, \dots).$$

Only pure morphisms should slide.

THEOREM. If  $\mathbb{V}$  is cartesian, and  $\mathcal{C}$  is weakly cartesian then this final coalgebra is constructed by observational streams.

EXTRA : EXPRESSIVITY

# EXPRESSIVITY

Orthogonal to the rest of the paper. What is the expressivity of the feedback syntax over some generators? The  $\text{St}(\cdot)$  construction answers this.

- $k$ -Linear functions: hidden multivariable linear recurrence equations.

$$\begin{pmatrix} s_1 \\ \vdots \\ s_n \\ \hline b_{t+1} \end{pmatrix} = \begin{pmatrix} \lambda_{11} & \dots & \lambda_{1n} & \mu_{11} & \dots & \mu_{1k} \\ \vdots & & \vdots & \vdots & & \vdots \\ \lambda_{n1} & \dots & \lambda_{nn} & \mu_{n1} & \dots & \mu_{nk} \end{pmatrix} \begin{pmatrix} s_1 \\ \vdots \\ s_n \\ \hline a_1 \\ \vdots \\ a_k \\ t \end{pmatrix}$$

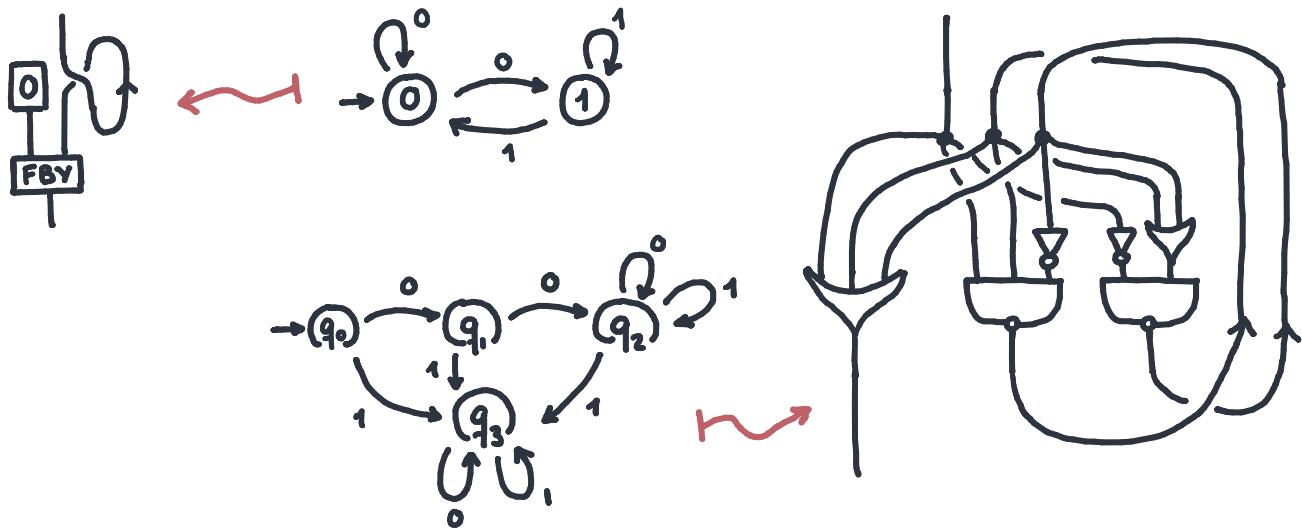
$$\begin{pmatrix} b_1 \\ \vdots \\ b_n \\ \hline t \end{pmatrix} = \begin{pmatrix} \psi_{11} & \dots & \psi_{1n} & \psi_{11} & \dots & \psi_{1k} \\ \vdots & & \vdots & \vdots & & \vdots \\ \psi_{n1} & \dots & \psi_{nn} & \psi_{n1} & \dots & \psi_{nk} \end{pmatrix} \begin{pmatrix} s_1 \\ \vdots \\ s_n \\ \hline a_1 \\ \vdots \\ a_k \\ t \end{pmatrix}$$

# EXPRESSIVITY

---

Orthogonal to the rest of the paper. What is the expressivity of the feedback syntax over some generators? The  $\text{St}(\cdot)$  construction answers this.

- Boolean circuits: controlled deterministic automata (w/ boolean IO).



# EXPRESSIVITY

Monoidal streams over total relations can be interpreted as Büchi automata.

$$\text{merge} : A + A \rightarrow A$$

$$\text{unreachable} : 0 \rightarrow A$$

$$\text{choose} : A \rightarrow A + A$$

$$\text{concat "x"} : A \rightarrow A$$

$$\text{token} : A \rightarrow A$$

EXAMPLE:  $0^*1(0^*11)^\omega$

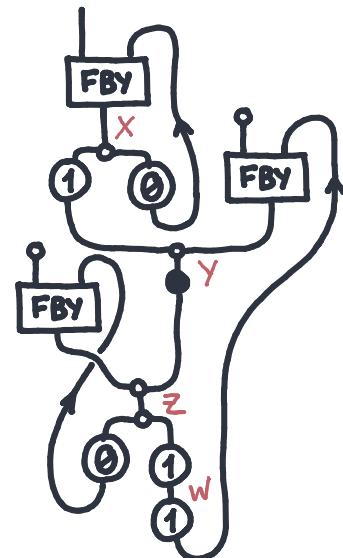
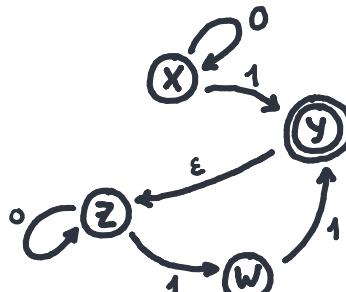
Park's equations.

$$X \Rightarrow 0X + 1Y$$

$$Y \Rightarrow Z$$

$$Z \Rightarrow 0Z + 1W$$

$$W \Rightarrow 1Y$$



Monoidal category: TOTALREL, +.

EXTRA : COINDUCTION

# COINDUCTIVE MONOIDAL STREAMS

---

DEFINITION (DLdFR). A monoidal stream  $f \in \text{Stream}(X_0, X_1, \dots; Y_0, Y_1, \dots)$  is

- a memory  $M(f) \in \mathbb{C}$
- a  $\text{now}(f) : X_0 \rightarrow M(f) \otimes Y_0$ ,
- and a  $\text{later}(f) \in \text{Stream}(M \otimes X_1, X_2, \dots; Y_1, Y_2, \dots)$ .

Quotiented by  $f \approx g$ , meaning

- the existence of  $r : M(f) \rightarrow M(g)$ ,
- such that  $\text{now}(f); (r \otimes \text{id}) = \text{now}(g)$ ,
- and such that  $\text{later}(f) \approx r \cdot \text{later}(g)$ .

# COINDUCTIVE MONOIDAL STREAMS

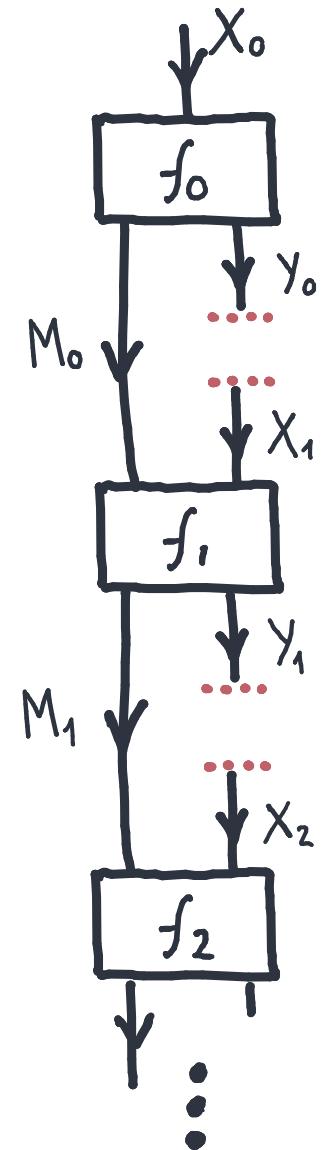
Reasoning with monoidal streams is easy: they are a final coalgebra.

$$S(X, Y) \cong \int^{M:C} \text{hom}(X_0, M \otimes Y_0) \times S(M \otimes X_0, X_1, X_2, \dots; Y_0, Y_1, Y_2, \dots).$$

Let me write  $X \in [N, C]$  for  $X_0, X_1, \dots$ ; write  $X^+$  for  $X_1, X_2, \dots$ ; and write  $M \cdot X$  for  $M \otimes X_0, X_1, X_2, \dots$ ; the equation becomes

$$S(X, Y) \cong \int^{M:C} \text{hom}(X_0, M \otimes Y_0) \times S(M \cdot X^+; Y^+).$$

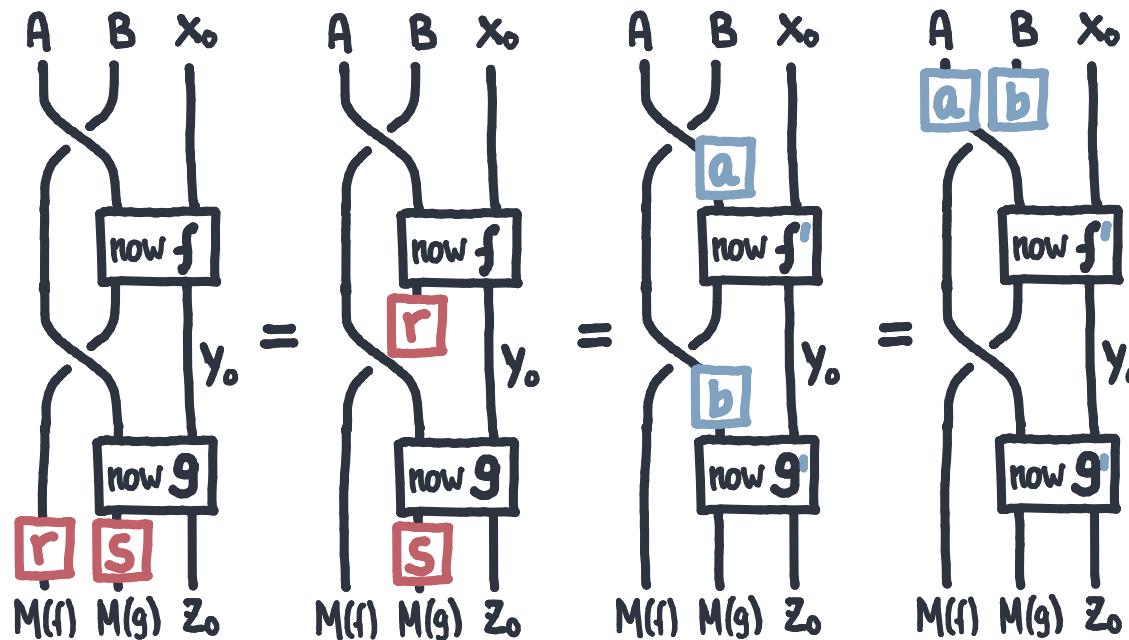
The coend connects the  $M$  to the next step.



# COINDUCTIVE MONOIDAL STREAMS

---

SEQUENTIAL COMPOSITION is well-defined. Given generators  $f \approx^r a \cdot f'$  and  $g \approx^s b \cdot g'$ , we can show that  $(f^A; g^B) \stackrel{r \otimes s}{\approx} (a \otimes b) \cdot (f'^A; g'^B)$ .



By coinduction,

$$\text{later}(f)^{M(f)} \circ \text{later}(g)^{M(g)} \underset{\text{(r}\otimes\text{s)}}{\approx} (\text{r}\otimes\text{s}) \cdot (\text{later}(f') \circ \text{later}(g'))^{M(f') \cup M(g')}$$

using  $\text{later}(f) \approx r \cdot \text{later}(f')$   
 $\text{later}(g) \approx s \cdot \text{later}(g')$ .

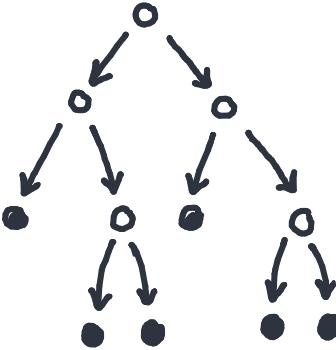
When  $a = \text{id}_A, b = \text{id}_B$ , we have  $(f \approx^r f') \wedge (g \approx^s g') \Rightarrow f^A; g^B \stackrel{r \otimes s}{\approx} f'^A; g'^B$ .

# ALGEBRA

---



Finite lists.  
 $L \cong A \times L + 1$



Finite trees.  
 $T \cong T^2 + 1$

1  
2  
3  
⋮

Natural numbers.  
 $N \cong N + 1$

Mathematics of syntax.

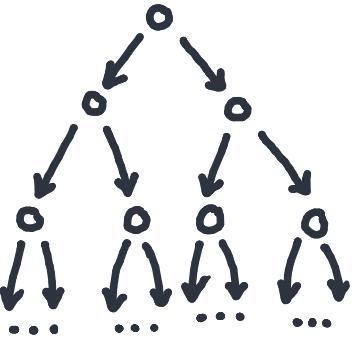
- Find the **initial** fixpoint of a functor,  $FX \cong X$ .
- Reason inductively.

# COALGEBRA

---



Infinite lists.  
 $L \cong A \times L$



Infinite trees.  
 $T \cong T^2$



Natural numbers.  
 $N \cong N + 1$

Algebra of state and transitions.

- Find the *final* fixpoint of a functor,  $FX \cong X$ .
- Reason coinductively.

# COALGEBRA

THEOREM (LAMBEK). If the **final coalgebra** exists, it is a **final fixpoint**.

THEOREM (ADAMEK). If the following limit is a fixpoint, it is final.

$$\lim_{n \in \mathbb{N}} (1 \leftarrow F1 \xleftarrow{F!} FF1 \xleftarrow{FF!} FFF1 \leftarrow \dots).$$

$$\begin{array}{ccc} X & \xrightarrow{f} & U \\ \alpha \downarrow & \parallel & \downarrow \cong \\ FX & \xrightarrow[Ff]{} & FU \end{array}$$

That is, to compute a fixpoint, repeatedly apply  $F$  and it will converge. Hopefully you will arrive to the fixpoint.

Initial algebras work too, but they will be less interesting.

# COINDUCTION

Reverse an infinite tree.

$$\text{reverse}(a; l, r) := (a, \text{reverse}(l), \text{reverse}(r))$$

$$\text{rev} \left( \begin{array}{c} \circ \\ \Delta \quad \Delta \end{array} \right) = \text{rev}(\Delta) \text{ rev}(\Delta)$$

Reverse is self-inverse.

Coinduction  
Hypothesis

$$\text{rev} \left( \text{rev} \left( \begin{array}{c} \circ \\ \Delta \quad \Delta \end{array} \right) \right) = \text{rev} \left( \text{rev}(\Delta) \text{ rev}(\Delta) \right) = \text{revrev}(\Delta) \text{ revrev}(\Delta) = \begin{array}{c} \circ \\ \Delta \quad \Delta \end{array} .$$

# EXTRA : OBSERVATIONAL STREAMS

# PROCESS INTERPRETATION

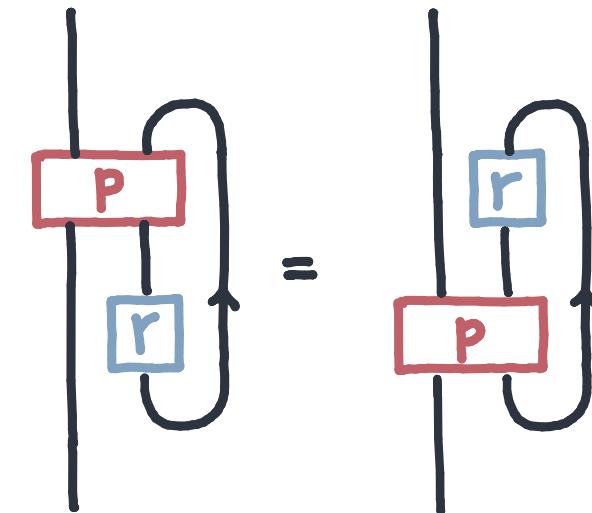
---

We think of functors  $\mathcal{C}^{\text{op}} \times \mathcal{C} \rightarrow \text{SET}$  as indexing families of processes,  $P(M, N)$ , by **input M** and **output N**. How to plug the output to the input?

Given  $r: N \rightarrow M$ ,

- $P(\text{id}, r)(p) \in P(M, M)$ , translate **after reading**,
- $P(r, \text{id})(p) \in P(N, N)$ , translate **before writing**.

These are "morally the same": **dinaturally equivalent**.



We write the set  $\int^{x \in \mathcal{C}_{\text{obj}}} P(x, x)$  for  $\sum_{x \in \mathcal{C}_{\text{obj}}} P(x, x) / \sim$ , the **coend** of  $P$ ,

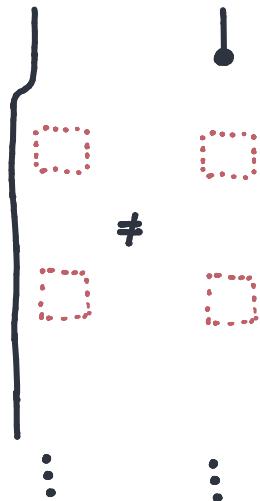
- $P(\text{id}, r)(p) \sim P(r, \text{id})(p)$ .

# OBSERVATIONAL EQUIVALENCE

---

So, we want, at least, extensional equivalence. Can we refine it?

Saving to memory without outputting is, observationally, the same as discarding. However, no amount of sliding will help us equating these two.

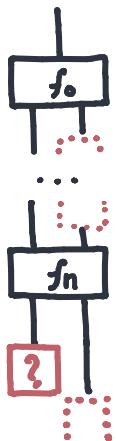


# OBSERVATIONAL EQUIVALENCE

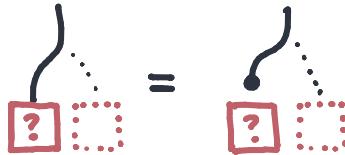
---

So, we want, at least, extensional equivalence. Can we refine it?

**DEFINITION.** The  $n$ th-truncation of an extensional stream  $\langle f_n \rangle$  is the first  $n$  components, up to any continuation.



Two streams are **observationally equal** if their  $n$ th truncations are equal. For instance,



This is not only a reasonable-sounding rule. This makes observational streams a canonical fixpoint.

# OBSERVATIONAL STREAMS

---

Intensional streams were the canonical fixpoint of the equation

$$T(X, Y) \cong \sum_{M \in C} \text{hom}(X_0, M \otimes Y_0) \times T(M \otimes X_1, X_2, X_3, \dots; Y_1, Y_2, Y_3, \dots).$$

**THEOREM (DLdFR).** *Observational streams*, in reasonably well-behaved categories (**productive**) are the canonical fixpoint of the equation

$$Q(X, Y) \cong \int^{M \in C} \text{hom}(X_0, M \otimes Y_0) \times Q(M \otimes X_1, X_2, X_3, \dots; Y_1, Y_2, Y_3, \dots).$$

Why not in all categories? You could craft a category where there is no way to go from  $X_0$  to  $Y_0$  without knowing the future!

I.e. there are infinitely descending chains for a Loebner-like order  $\frac{\sqsupseteq}{\sqsubset} = \frac{\sqsupseteq_{fin}}{\sqsubset}$ .

# OBSERVATIONAL STREAMS

Why not in **all categories**? You could craft a category where there is no way to go from  $X_0$  to  $Y_0$  without knowing the future! Adamek could fail.

$$\int^{M:C} \text{hom}(X_0, M \otimes Y_0) \times \lim_{n \in N} \int^{M_1, \dots, M_n} \prod_{i=1}^n \text{hom}(M_{i-1} \otimes X_i, M_i \otimes Y_i)$$

$\cong$  ✓ fine

$$\int^{M:C} \lim_{n \in N} \int^{M_1, \dots, M_n} \text{hom}(X_0, M \otimes Y_0) \times \prod_{i=1}^n \text{hom}(M_{i-1} \otimes X_i, M_i \otimes Y_i)$$

$\cong$  ✗ only discrete coproducts commute with connected limits

$$\lim_{n \in N} \int^{M:C} \int^{M_1, \dots, M_n} \text{hom}(X_0, M \otimes Y_0) \times \prod_{i=1}^n \text{hom}(M_{i-1} \otimes X_i, M_i \otimes Y_i)$$

I.e. there are infinitely descending chains for a Loebner-like order  $\frac{\square f_n}{\square} = \frac{\square f_{n+1}}{\square}$ .

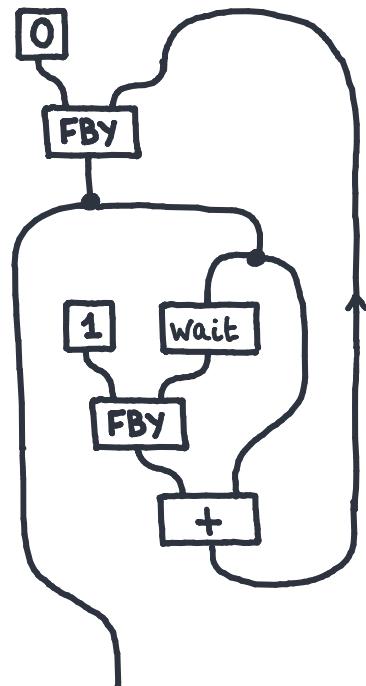
Nothing to worry in semi cartesian, compact closed and freely generated monoidal.

EXTRA : EXAMPLE

# MOTIVATION: DATAFLOW PROGRAMMING

$\text{fib} = 0 \text{ FBY } (\text{fib} + 1 \text{ FBY } \text{WAIT fib})$

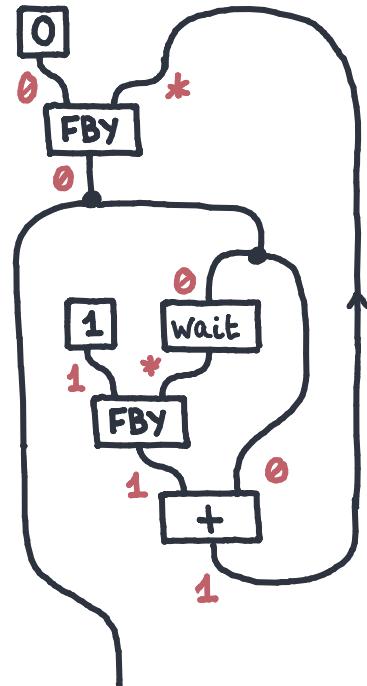
Time.	$\text{fib}$	$\text{WAIT(fib)}$	$1 \text{FBYWAIT fib}$
0			
1			
2			
3			
4			
:			



# MOTIVATION: DATAFLOW PROGRAMMING

$\text{fib} = 0 \text{ FBY } (\text{fib} + 1 \text{ FBY } \text{WAIT fib})$

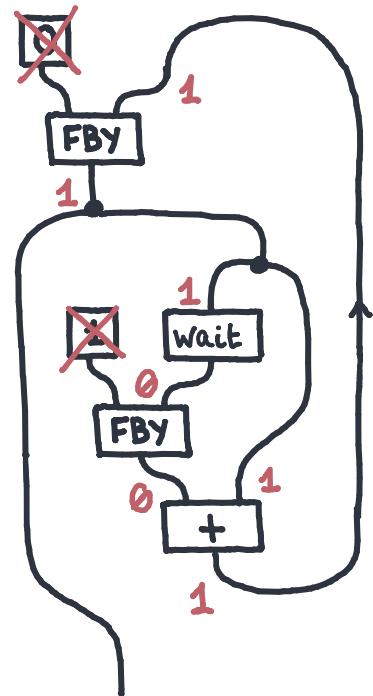
Time.	$\text{fib}$	$\text{WAIT(fib)}$	$1 \text{FBYWAIT fib}$
0	0	*	
1			1
2			
3			
4			
:			



# MOTIVATION: DATAFLOW PROGRAMMING

$\text{fib} = 0 \text{ FBY } (\text{fib} + 1 \text{ FBY } \text{WAIT fib})$

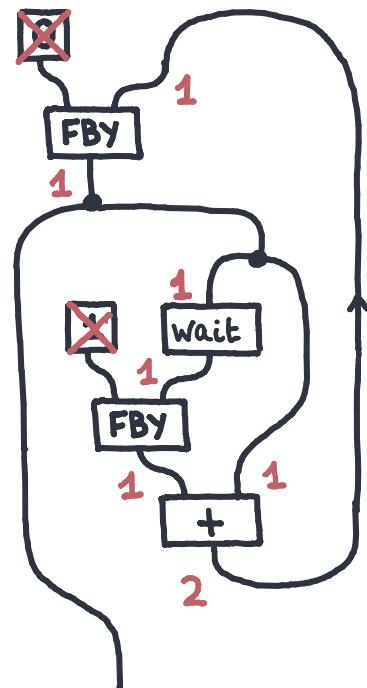
Time.	$\text{fib}$	$\text{WAIT(fib)}$	$1_{\text{FBYWAIT fib}}$
0	0	*	1
1	1	0	0
2			
3			
4			
:			



# MOTIVATION: DATAFLOW PROGRAMMING

$\text{fib} = 0 \text{ FBY } (\text{fib} + 1 \text{ FBY } \text{WAIT fib})$

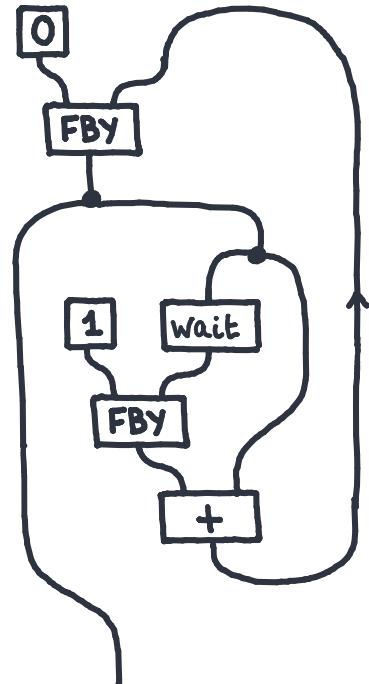
Time.	$\text{fib}$	$\text{WAIT(fib)}$	$1_{\text{FBYWAIT fib}}$
0	0	*	1
1	1	0	0
2	1	1	1
3			
4			
:			



# MOTIVATION: DATAFLOW PROGRAMMING

$\text{fib} = 0 \text{ FBY } (\text{fib} + 1 \text{ FBY } \text{WAIT fib})$

Time.	$\text{fib}$	$\text{WAIT(fib)}$	$1 \text{FBYWAIT fib}$
0	0	*	1
1	1	0	0
2	1	1	1
3	2	1	1
4	3	2	2
:	:	:	:

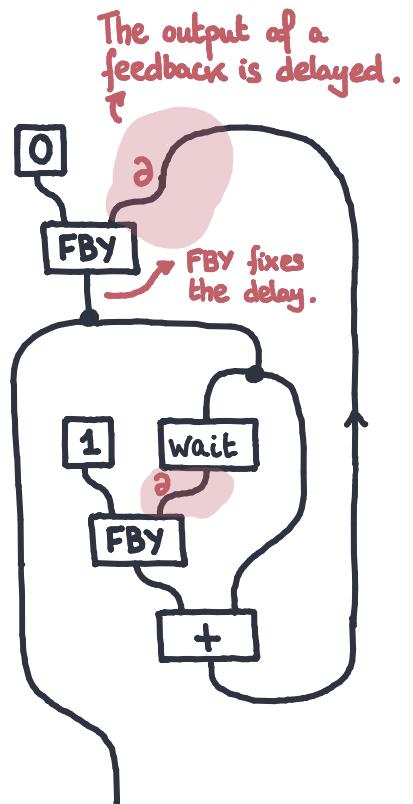


# MOTIVATION: DATAFLOW PROGRAMMING

$\text{fib} = 0 \text{ FBY } (\text{fib} + 1 \text{ FBY } \text{WAIT fib})$

Time.	$\text{fib}$	$\text{WAIT(fib)}$	$1\text{FBYWAIT fib}$
0	0	*	1
1	1	0	0
2	1	1	1
3	2	1	1
4	3	2	2
:	:	:	:

How to ensure the output is well-defined? **Delayed types.**



EXTRA : LENSES

---

# DINATURALITY AND COENDS

---

Given  $P: \mathcal{C}^{\text{op}} \times \mathcal{C} \rightarrow \text{SET}$ , consider the set of all processes.  $\sum_{x \in \mathcal{C}_{\text{obj}}} P(x, x)$ .  
**Dinatural equivalence** is the smallest equivalence relation generated by

$$P(\text{id}, r)(p) \sim P(r, \text{id})(p).$$

We write the set  $\int^{x \in \mathcal{C}_{\text{obj}}} P(x, x)$  for  $\sum_{x \in \mathcal{C}_{\text{obj}}} P(x, x)/\sim$ , the **coend** of  $P$ .

“The coend,  $\int^{x \in \mathcal{C}_{\text{obj}}}$ , connects both  $X$ 's.”

# MOTIVATION

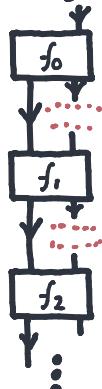
---

In the process interpretation of monoidal categories, morphisms  $A \rightarrow B$  are processes with an **input A** and an **output B**.

However, most processes (servers, drivers, agents,...) are continuously taking inputs and producing outputs,



closed diagram vs



open<sup>[1]</sup>, repeated<sup>[2]</sup> diagram.



Román.<sup>[1]</sup> Open Diagrams via Coend Calculus. ACT'20.

Román.<sup>[2]</sup> Comb Diagrams for Discrete-Time Feedback. Preprint.

# ADÁMEK'S THEOREM

---

**THEOREM.** Let  $\mathcal{C}$  be a category with a terminal object and let  $F$  be any endofunctor  $F: \mathcal{C} \rightarrow \mathcal{C}$ . If the following limit

$$L \cong \lim_{n \in \mathbb{N}} (1 \leftarrow F1 \xleftarrow{F!} FF1 \xleftarrow{FF!} FFF1 \leftarrow \dots)$$

exists and is preserved by  $F$ , then it carries a final coalgebra structure.

That is, to compute a fixpoint, repeatedly apply  $F$  and it will converge. Hopefully you will arrive to the fixpoint.

Initial algebras work too, but they will be less interesting in our case.

# MOTIVATION

---

Lenses can be used to describe a single exchange,

$$\text{Lenses}(A_0, A_1; B_0, B_1) = \int^M \text{hom}(A_0, B_0 \times M) \times \text{hom}(M \times A_1, B_1) \cong \text{hom}(A_0, B_0) \times \text{hom}(A_0 \times A_1, B_1);$$

but "fixpointing" the exchange, we get causal stream functions.

$$\text{Stream}(A; B) = \int^M \text{hom}(A_0, B_0 \times M) \times \text{Stream}(M \cdot A^+; B^+) = \text{hom}(A_0, B_0) \times \text{Stream}(A_0; A^+; B^+).$$

$$\text{Stream}(A; B) \cong \text{Optic}(\text{Set}, \text{Stream})((A_0, A^+); (B_0, B^+)).$$

Given the category  $\text{Stoch}$  of stochastic functions, the category of stochastic processes is the terminal fixpoint of

$$\text{StochProc}(A; B) \cong \text{Optic}(\text{Stoch}, \text{StochProc})((A_0, A^+); (B_0, B^+))$$

# CAUSAL STREAM FUNCTIONS

DEFINITION. A *causal stream function*  $f: \mathbb{X} \rightarrow \mathbb{Y}$  is a family of functions

$$f_n: X_0 \times \dots \times X_n \rightarrow Y_n.$$

$$f_0: X_0 \rightarrow Y_0$$

$$f_1: X_0 \times X_1 \rightarrow Y_1$$

$$f_2: X_0 \times X_1 \times X_2 \rightarrow Y_2$$

...

These form a monoidal category, the cokleisli category of the non-empty list monoidal comonad,

$$\text{List}^+: [\mathbf{N}, \mathbf{SET}] \rightarrow [\mathbf{N}, \mathbf{SET}], \text{List}^+(\mathbb{X})_n = \prod_{i=0}^n X_i.$$

We have

- a delay functor taking  $\mathbb{X} = (X_0, X_1, X_2, \dots)$  into  $\partial\mathbb{X} = (1, X_0, X_1, \dots)$ ;
- a trace-like operator taking  $\partial S \circ \mathbb{X} \rightarrow S \circ \mathbb{Y}$  into  $\mathbb{X} \rightarrow \mathbb{Y}$ ;
- coalgebraic reasoning and coinductive arguments.



# SYNOPSIS

---

Three definitions from universal properties, and three explicit constructions.  
Each one a quotient of the previous.

1. Intensional streams, a first naïve version. Fail to form a category.
2. Extensional streams, a free category with feedback.
3. Observational streams, definitive solution to a fixpoint equation.

Two known particular cases, and an avenue for more.

1. Cartesian monoidal streams  $(\text{Set}, \times)$  are causal functions  
(as in Uustalu-Vene, Sprunger-Jacobs).
2. Stochastic streams  $(\text{Kl}(\mathcal{D}), \times)$  are controlled stochastic processes  
(classical in the literature).
3. Kleisli streams of strong monads. Freyd categories in general.

# OTHER THEORIES OF PROCESSES

---

These will be monoidal streams.

We imagine what the syntax should be, but, what should replace streams in the semantics?

THEOREM (DLdFR) The non-empty list functor

$$\text{List}^+(\mathbb{X})_n = \bigotimes_{i=0}^n \mathbb{X}_i$$

is a monoidal comonad if and only if  $(\otimes)$   
is a cartesian product.

So far, so good with deterministic functions.