Geometry of Interaction and higher order functions

Naohiko Hoshino
Kyoto University
(Mathematical logic and its Application 2016)

My research area

(Functional) programming language

- proving properties of programs
- designing programming languages
- exploring programming techniques

using mathematical models of programming languages

$$\text{programs} \longmapsto \text{mathematical objects}$$

This talk: Geometry of Interaction [Girard '89]

Computable Functions

A partial function $f\colon \mathbf{N} \rightharpoonup \mathbf{N}$ is computable iff

- f is definable by means of a Turing machine
- f is a recursive function
- f is representable by an untyped lambda term

How about higher order functions?

PCF

[Plotkin '77]

Programming language for Computatble Functions based on Scott's LCF (Logic of Computable Functions)

- The simply typed lambda calculus
  - $+$ Function application $(f, x) \longmapsto f(x)$
  - $+$ Currying $\lambda x.f(x, y)$
- Natural numbers
  - $+$ $0, 1, 2, \ldots$
  - $+$ succ, pred, if-then-else
- Recursion on arbitrary types

Parallel Testing

Prop. There is no PCF-term

$$\mathbf{pconv} : \mathbf{Nat} \Rightarrow \mathbf{Nat} \Rightarrow \mathbf{Nat}$$

such that

- if $M \longrightarrow^* 0$ then $\mathbf{pconv}\, M\ N \longrightarrow^* 0$
- if $N \longrightarrow^* 0$ then $\mathbf{pconv}\, M\ N \longrightarrow^* 0$
- otherwise, $\mathbf{pconv}\, M\ N \longrightarrow^\infty$

Proof Use domain theory.

Remark. PCF+$\mathbf{pconv}$ is implementable as follows

$$\mathbf{pconv}\, M\ N \longrightarrow \mathbf{pconv}\, N\ M' \longrightarrow \cdots$$
$$\longrightarrow \mathbf{pconv}\, M'\ N' \longrightarrow \mathbf{pconv}\, 0\ N \longrightarrow 0$$

Checking Strictness

Prop. There is no PCF-term

$$\mathbf{strict}\colon (\mathbf{Nat} \Rightarrow \mathbf{Nat}) \Rightarrow \mathbf{Nat}$$

such that

- if $u(\Omega) \longrightarrow^* 0$ then $\mathbf{strict}(u) \longrightarrow^* 0$
- if $u(\Omega) \longrightarrow^\infty$ and $u(0) \longrightarrow^* 0$ then $\mathbf{strict}(u) \longrightarrow^* 1$
- otherwise, $\mathbf{strict}(u) \longrightarrow^\infty$

where $\Omega\colon \mathbf{Nat} \Rightarrow \mathbf{Nat}$ is defined by $\Omega\ x = \Omega\ x$.

Proof Use domain theory.

Remark. PCF$+\mathbf{strict}$ is implementable by checking whether evaluation of $u(0)$ touches 0.

$\mathbf{pconv}$ vs $\mathbf{strict}$

PCF$+\mathbf{pconv}+\mathbf{strict}$ is not implementable

Prop. There is no effective operational semantics for PCF$+\mathbf{pconv}+\mathbf{strict}$ such that

$$M \xrightarrow{\mathrm{PCF}} N \text{ iff } M \xrightarrow{\mathrm{extended}} N$$

for any PCF-term M

Proof For any $M : \mathbf{Nat}$, we can check termination of M by evaluating

$$\mathbf{strict}(\lambda x : \mathbf{Nat}.\, \mathbf{pconv}(x, \mathbf{if}\, M\, \mathbf{then}\, 0\, \mathbf{else}\, 0))$$

Extensions

PCF+$\mathbf{H}$ [Longley '02] PCF+$\mathbf{por}$+$\mathbf{exists}$ [Plotkin '77]

PCF+$\mathbf{strict}$+$\mathbf{gustave}$ [Paolini '06]

PCF+$\mathbf{strict}$ PCF+$\mathbf{pconv}$

PCF

Question

$\boxed{\text{How many extensions of PCF are there?}}$

- Given a candidate PCF+$\mathbf{foo}$, it is not easy to directly check that PCF+$\mathbf{foo}$ is really a new extension

  $\Longrightarrow$ categorical semantics is a powerful tool for checking definability

- Geometry of Interaction provides a recipe to generate mathematical models for (extensions) of PCF

Outline

The aim: explore diversity of Geometry of Interaction

1. Overview of Geometry of Interaction recipe
2. Three concrete SK-algebras based on the recipe
3. Main results: characterization of two categories in domain theory
   - Coherence spaces (PCF+$\mathbf{strict}$+$\mathbf{gustave}$)
   - Scott domains (PCF+$\mathbf{por}$+$\mathbf{exists}$)

(c.f. characterization of hypercoherence spaces (PCF+$\mathbf{H}$))

[Oosten '99],[Longley '02]

Geometry of Interaction

Recipe for SK-algebras [Abramsky, Haghverdi and Scott '02]

1. Choose a traced symmetric monoidal category $\mathcal{C}$
2. Apply $\mathbf{Int}$-construction
3. Solve a domain equation in $\mathbf{Int}(\mathcal{C})$

Then you will get an SK-algebra.

Def. An SK-algebra is a set A with a binary application and $S, K \in A$ such that

$$Sxyz = xz(yz) \qquad Kxy = x$$

Partial function

Def $f, g \colon \mathbf{N} \rightharpoonup \mathbf{N}$

$(f \cdot g)(x) = y$ iff $f(2x+1) = 2y+1$ or

$$\bigvee_{\vec{x},\vec{y}} \begin{array}{l} f(2x+1) = 2x_1 \ \& \ g(x_1) = y_1 \ \& \\ f([x, y_1]) = 2x_2 \ \& \ g(x_2) = y_2 \ \& \\ f([x, y_1, y_2]) = 2x_3 \ \& \ g(x_3) = y_3 \ \& \\ \quad\vdots \\ f([x, y_1, \ldots, y_n]) = 2y+1 \end{array}$$

Prop. $\mathbf{Pfn}(\mathbf{N}, \mathbf{N})$ is an SK-algebra.

Partial function

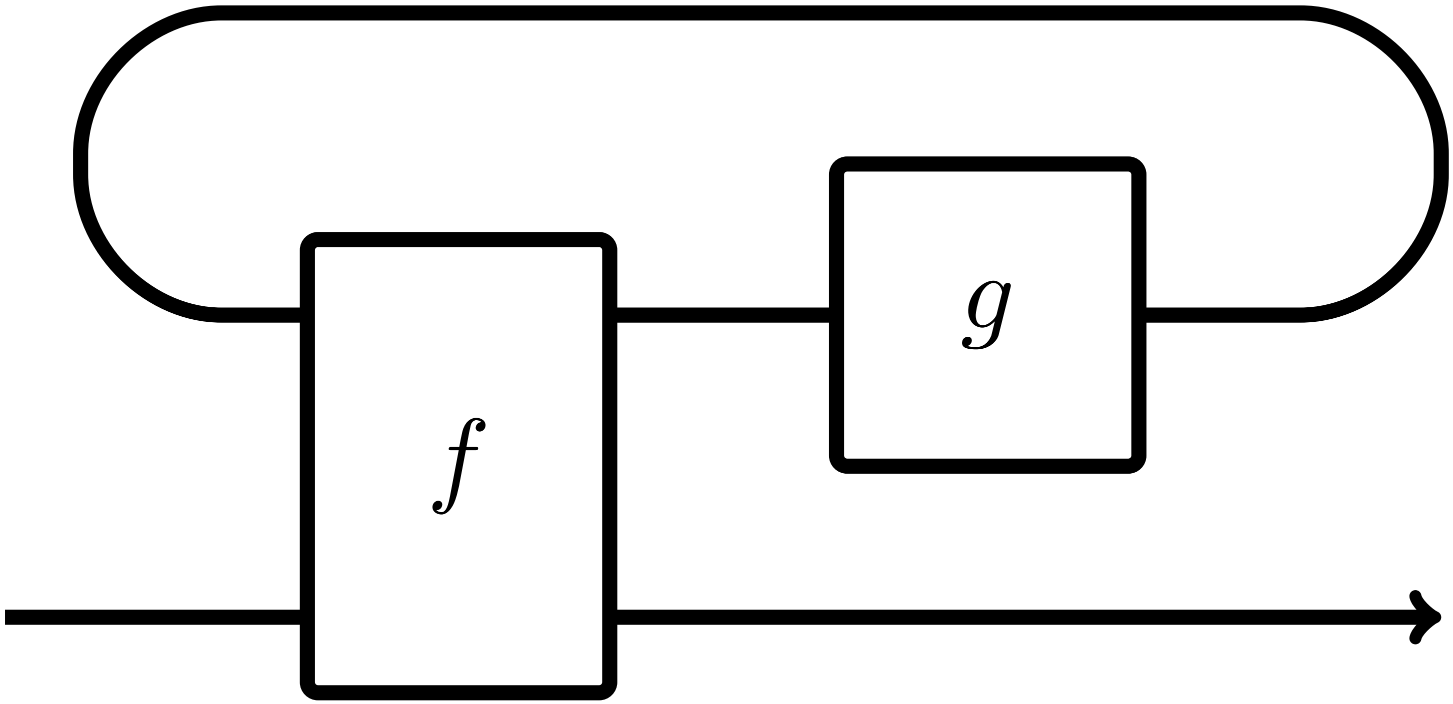
Def $f, g\colon \mathbf{N} \rightharpoonup \mathbf{N}$

$(f \cdot g)(x) = y$ iff $f(2x+1) = 2y+1$ or

$$\boxed{\mathbf{N}^* \cong 2\mathbf{N}} \qquad \bigvee_{\vec{x},\vec{y}} \begin{aligned} & f(2x+1) = 2x_1 \;\&\; g(x_1) = y_1 \;\& \\ & f(\underline{[x, y_1]}) = 2x_2 \;\&\; g(x_2) = y_2 \;\& \\ & f(\underline{[x, y_1, y_2]}) = 2x_3 \;\&\; g(x_3) = y_3 \;\& \\ & \quad \vdots \\ & f(\underline{[x, y_1, \ldots, y_n]}) = 2y+1 \end{aligned}$$

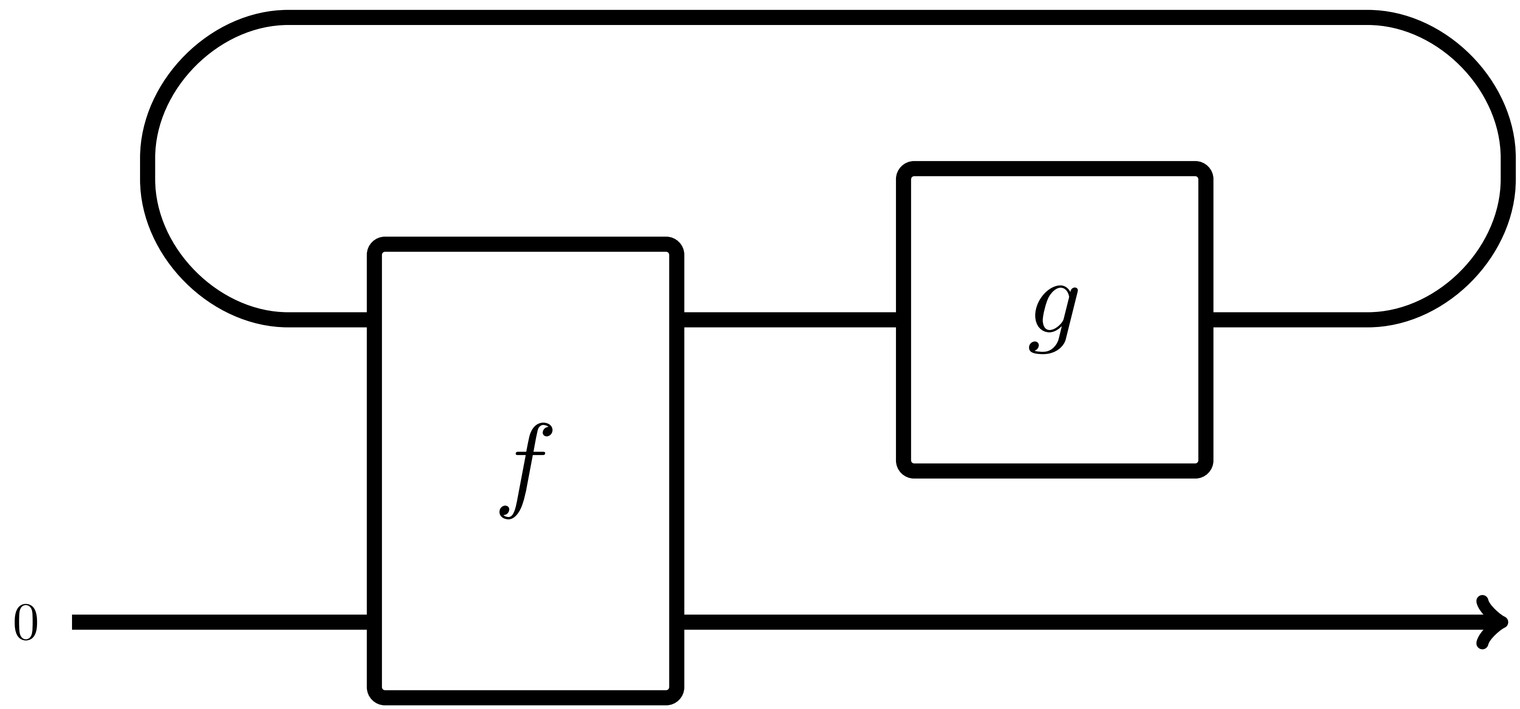
Prop. $\mathbf{Pfn}(\mathrm{N}, \mathrm{N})$ is an SK-algebra.

Interaction



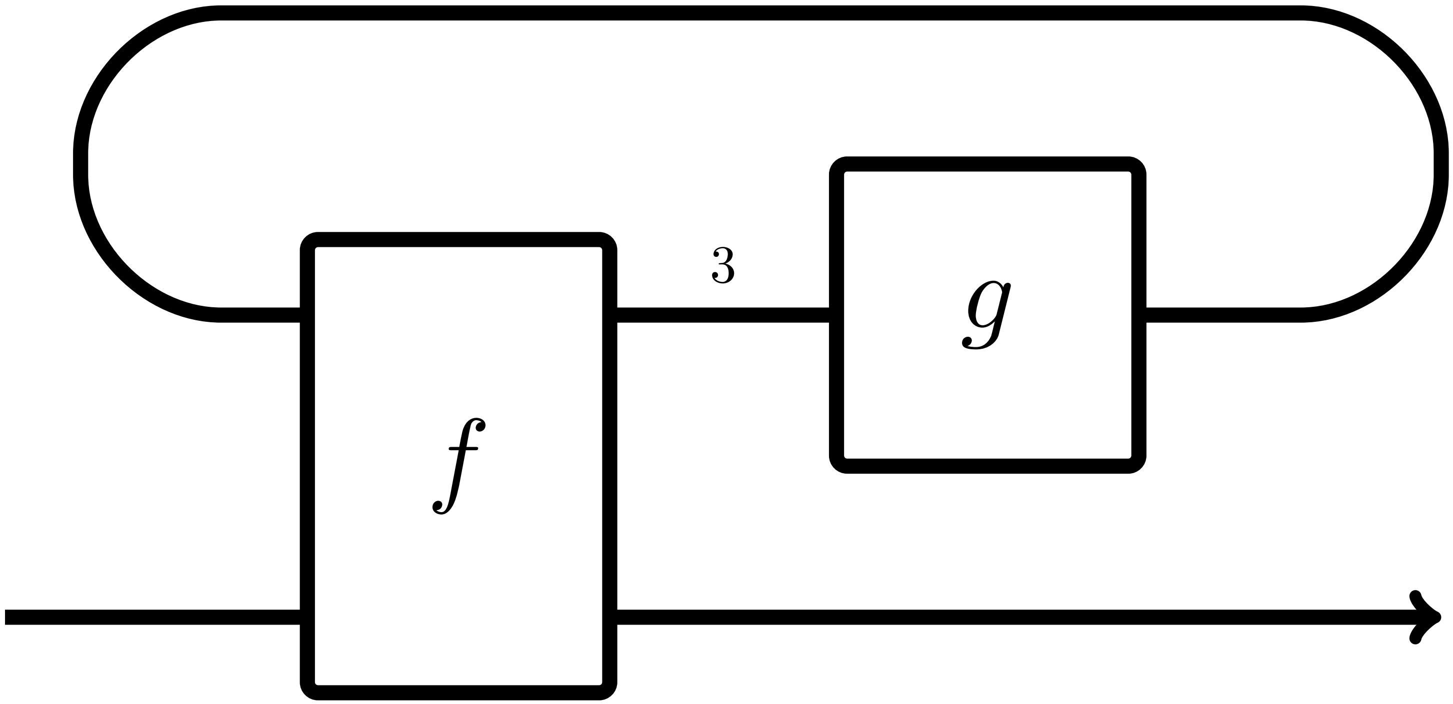
- f can remember interaction history while g can not
- both f and g behave deterministically

Interaction



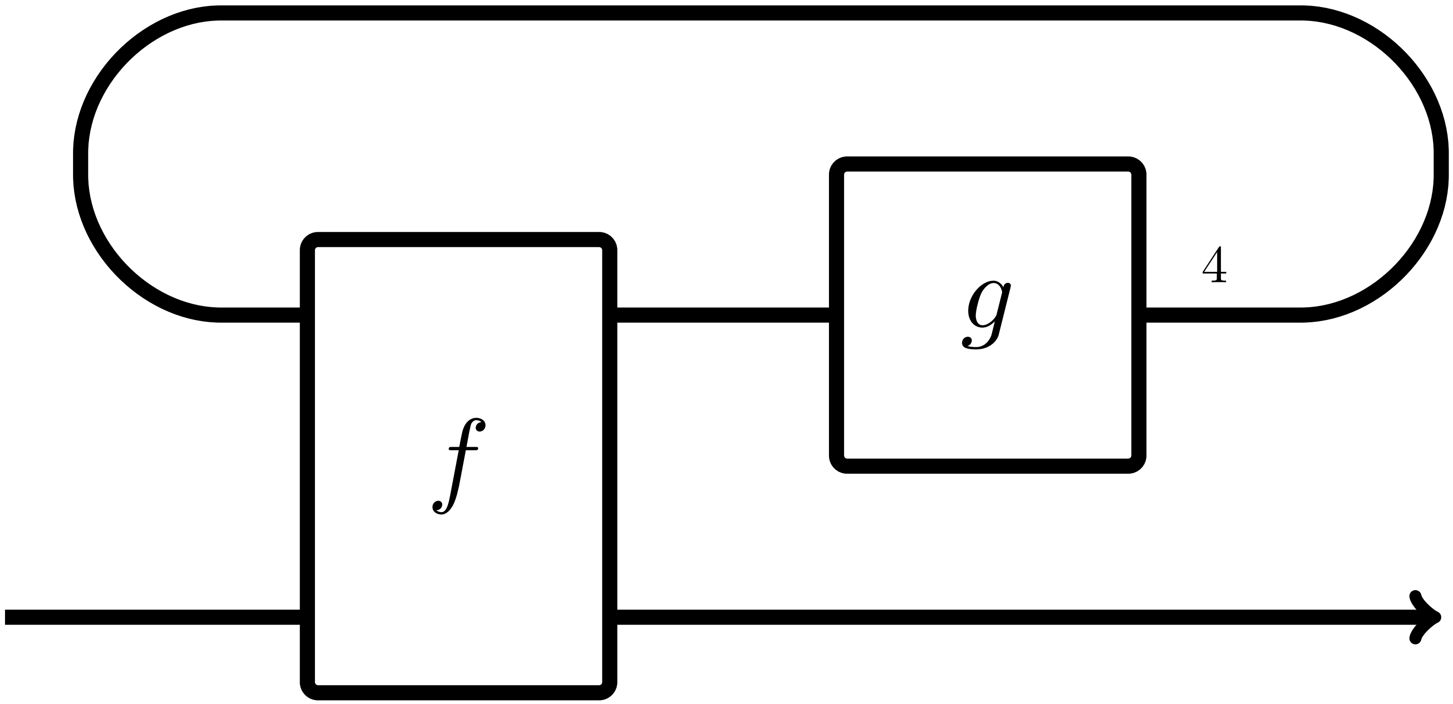
- f can remember interaction history while g can not
- both f and g behave deterministically

Interaction



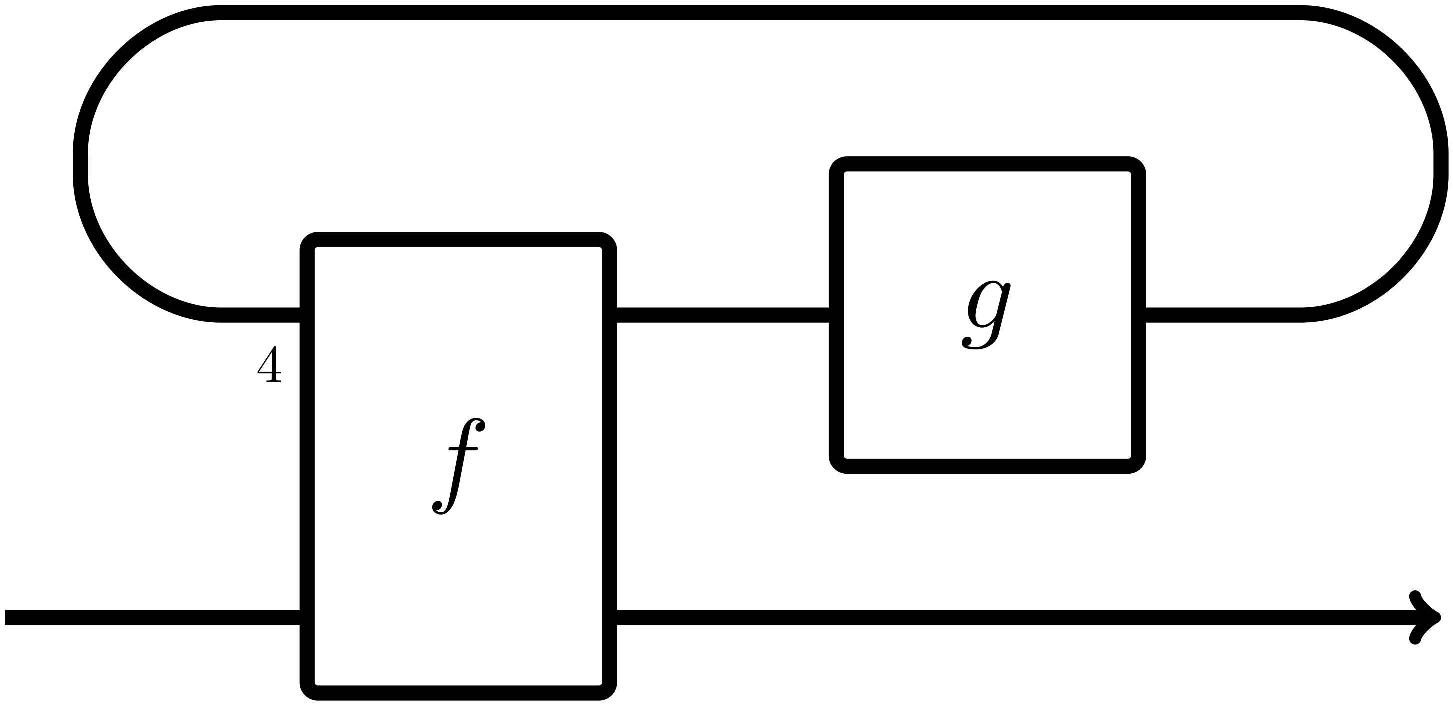
- f can remember interaction history while g can not
- both f and g behave deterministically

Interaction



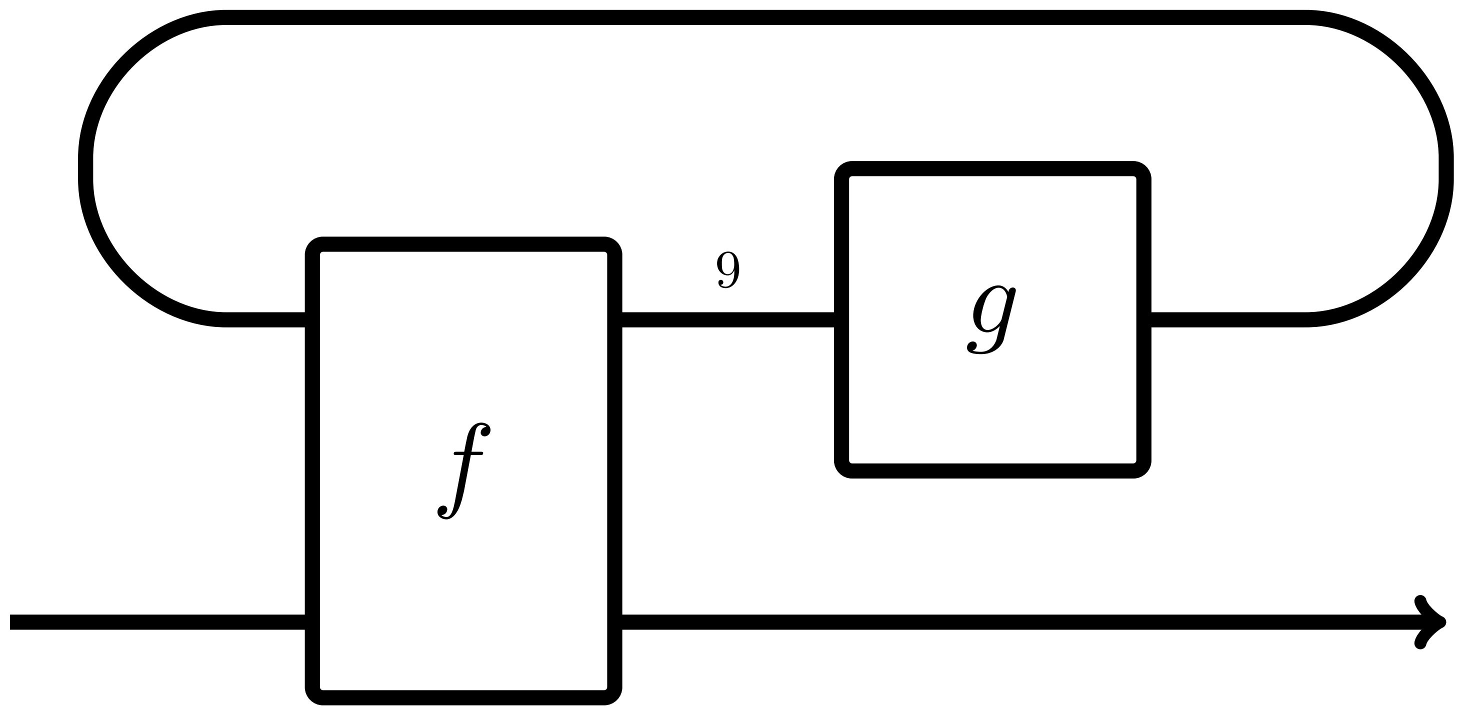
- f can remember interaction history while g can not
- both f and g behave deterministically

Interaction



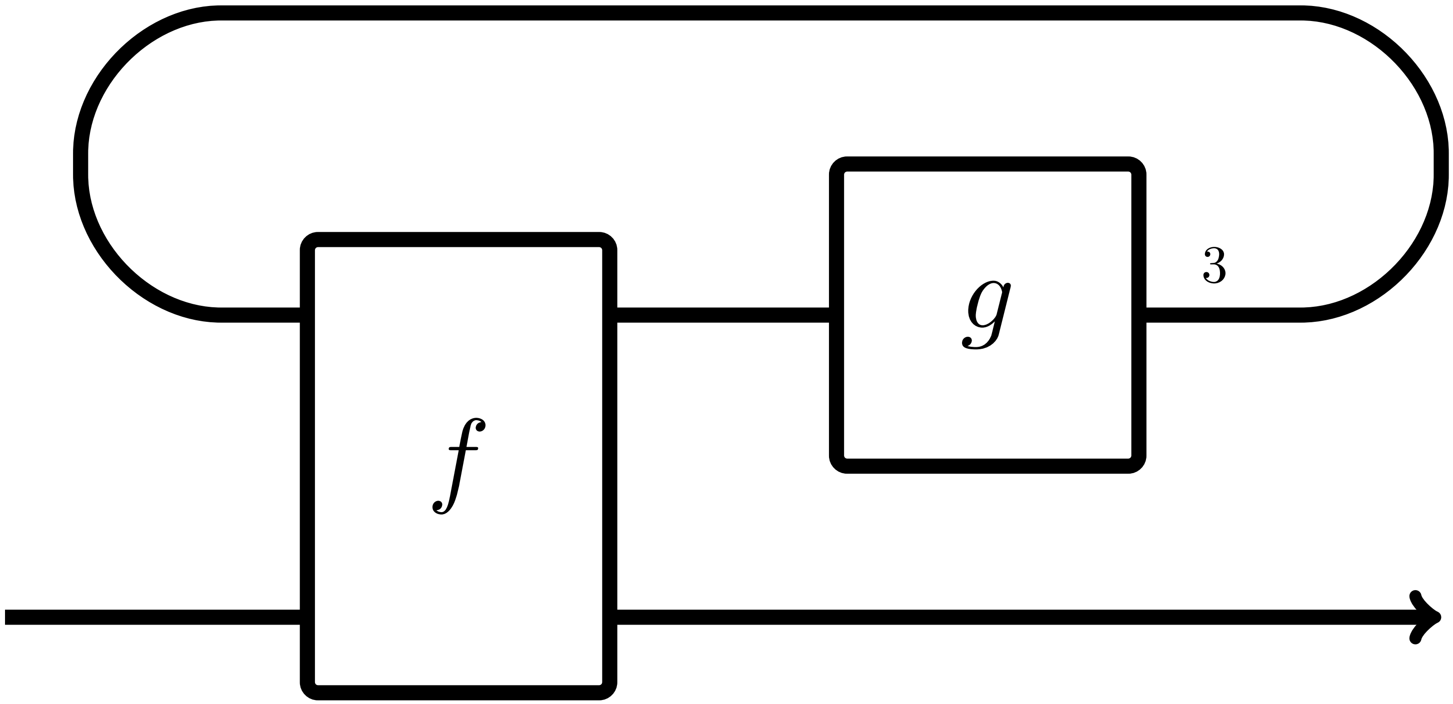
- f can remember interaction history while g can not
- both f and g behave deterministically

Interaction



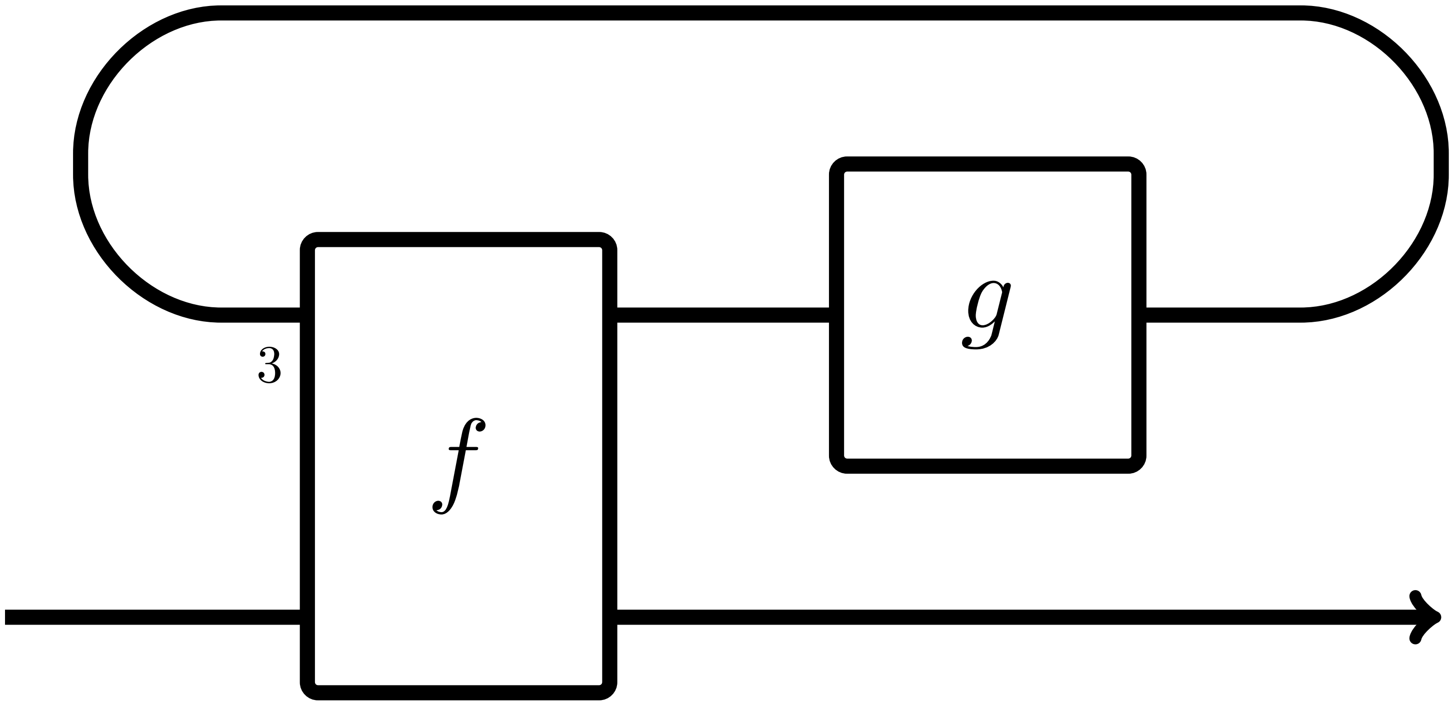
- f can remember interaction history while g can not
- both f and g behave deterministically

Interaction



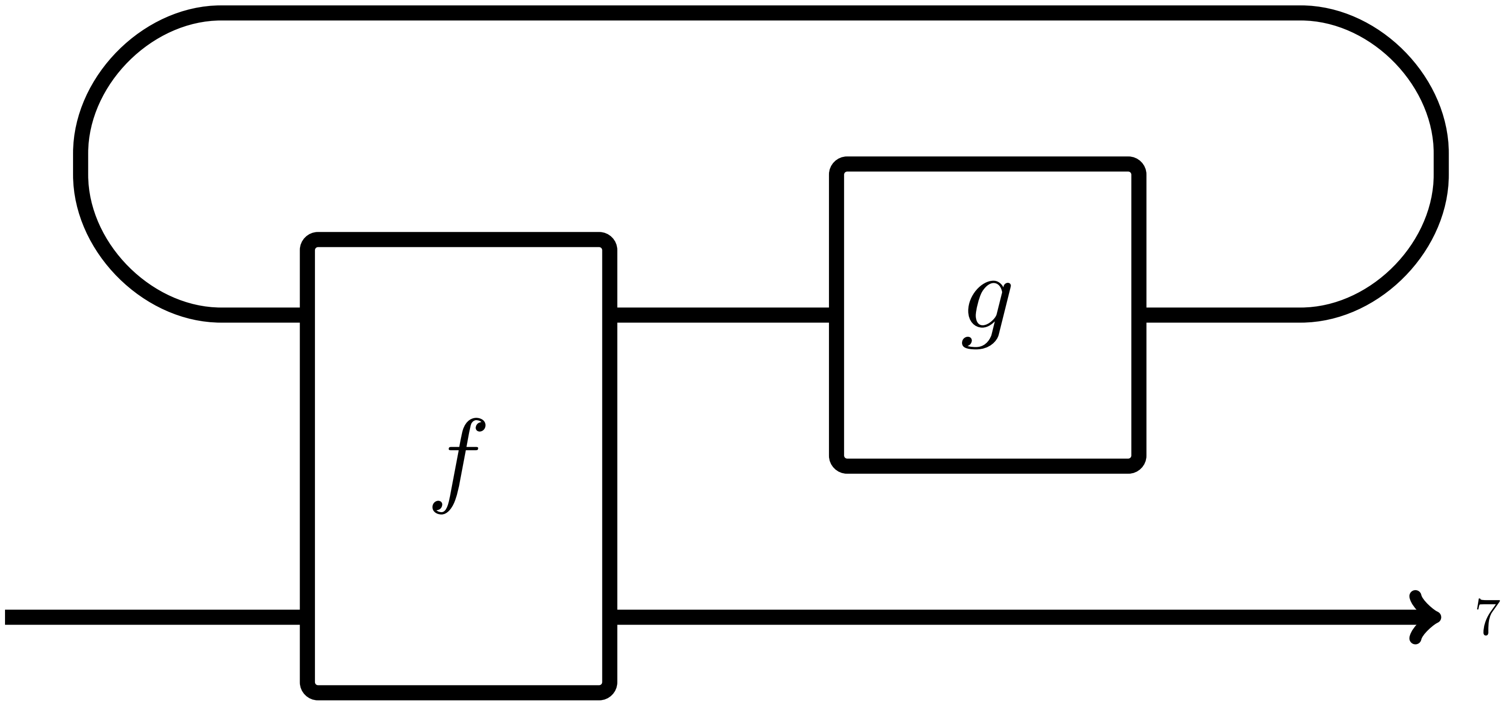
- f can remember interaction history while g can not
- both f and g behave deterministically

Interaction



- f can remember interaction history while g can not
- both f and g behave deterministically

Interaction



- f can remember interaction history while g can not
- both f and g behave deterministically

Relations

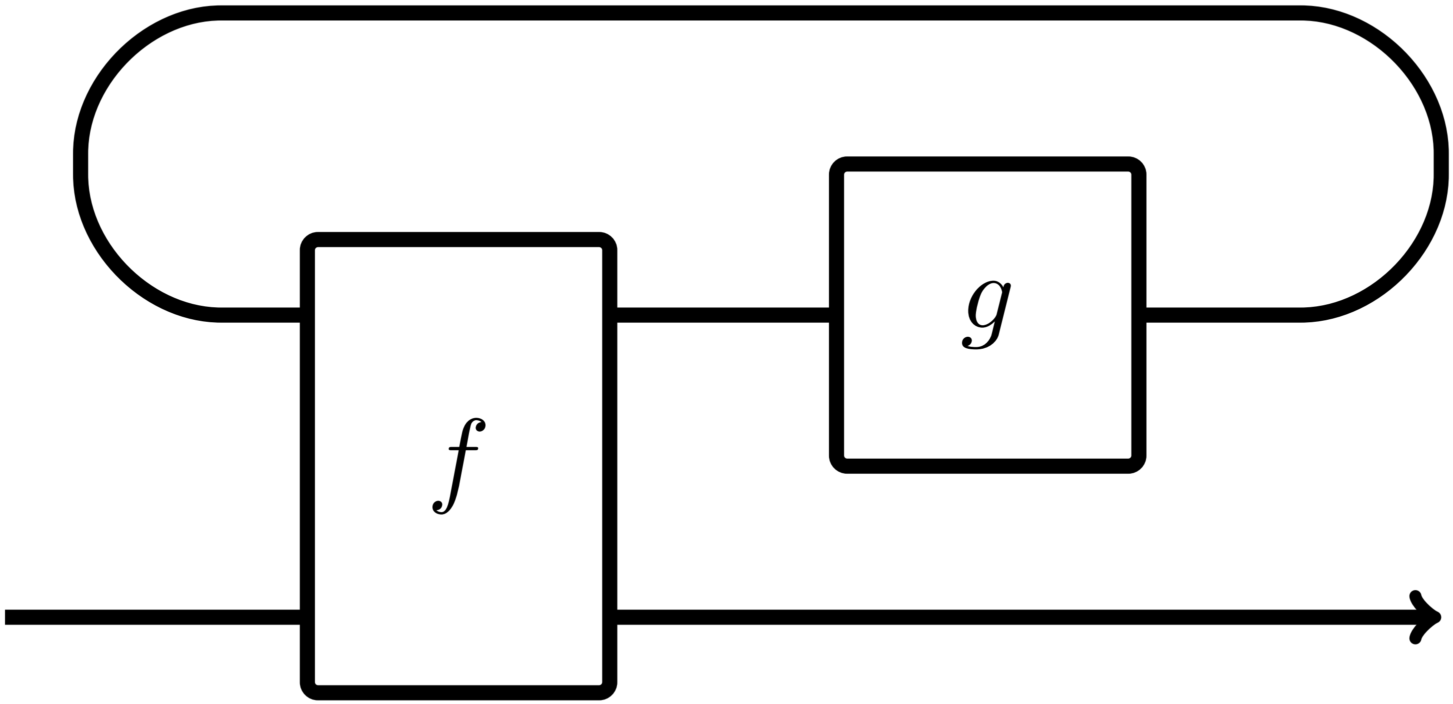
Def $f, g \colon \mathbf{N} \to 2^{\mathbf{N}}$

$(f \cdot g)(x) \ni y$ iff $f(2x+1) \ni 2y+1$ or

$$\bigvee_{\vec{x},\vec{y}} \begin{array}{l} f(2x+1) \ni 2x_1 \;\&\; g(x_1) \ni y_1 \;\&\; \\ f([x, y_1]) \ni 2x_2 \;\&\; g(x_2) \ni y_2 \;\&\; \\ f([x, y_1, y_2]) \ni 2x_3 \;\&\; g(x_3) \ni y_3 \;\&\; \\ \quad\vdots \\ f([x, y_1, \ldots, y_n]) \ni 2y+1 \end{array}$$

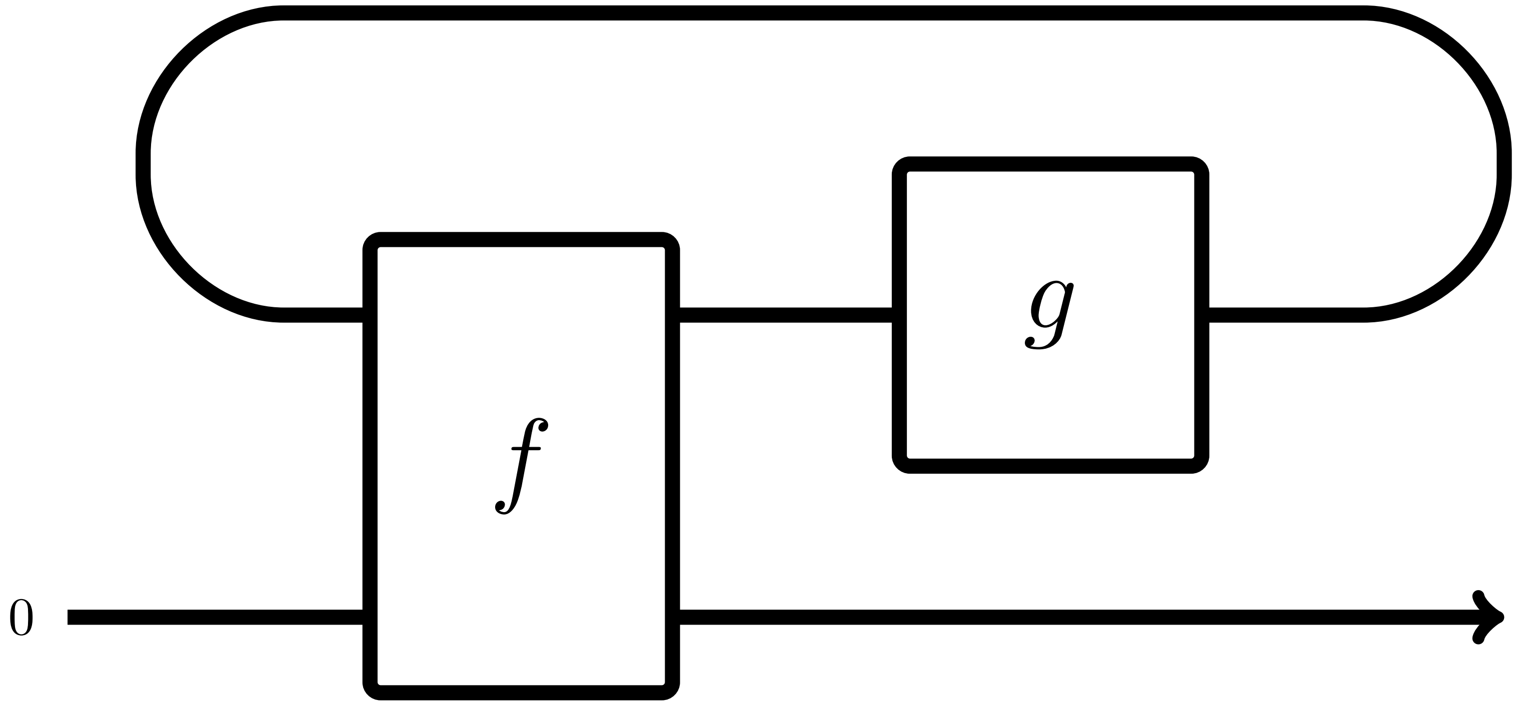
Prop. $\mathbf{Rel}(\mathbf{N}, \mathbf{N})$ is an SK-algebra.

Interaction



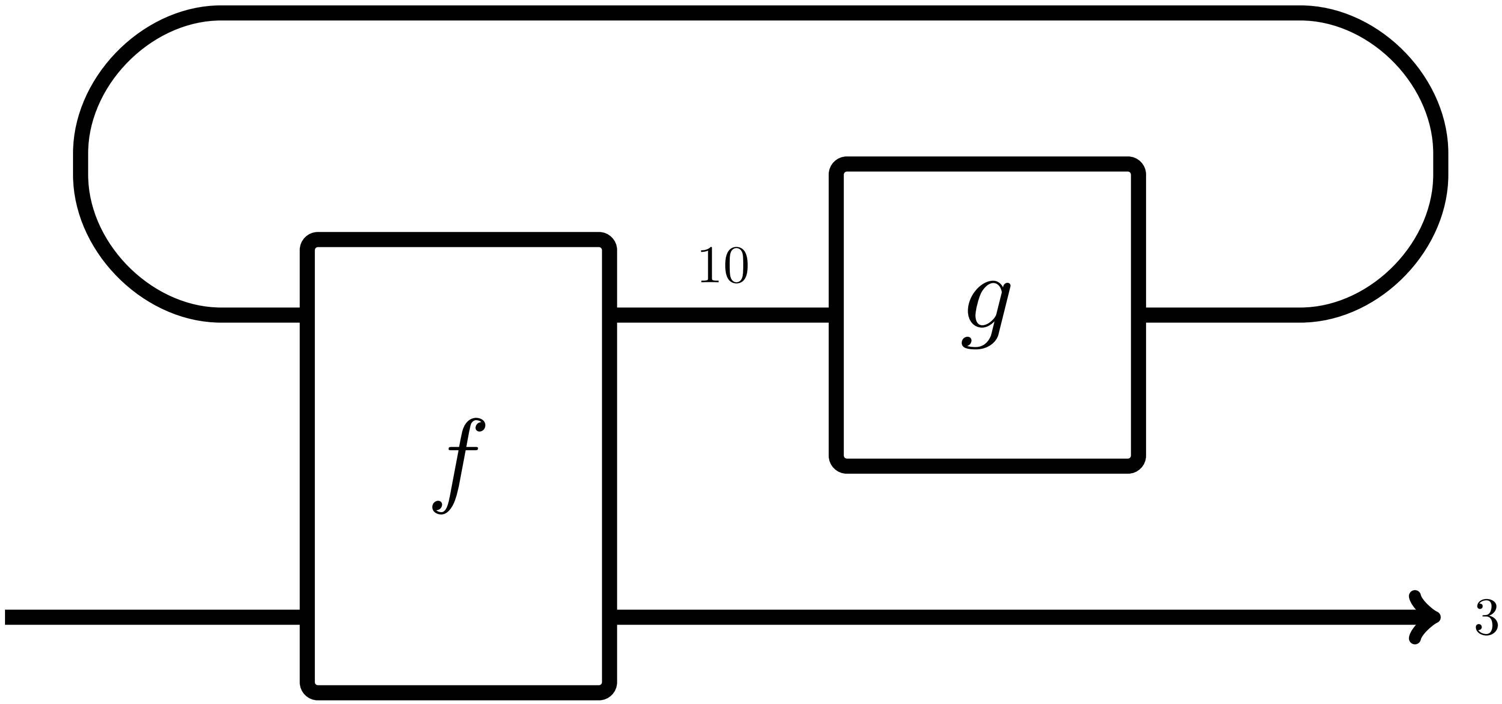
- f can remember interaction history while g can not
- f and g may behave nondeterministically

Interaction



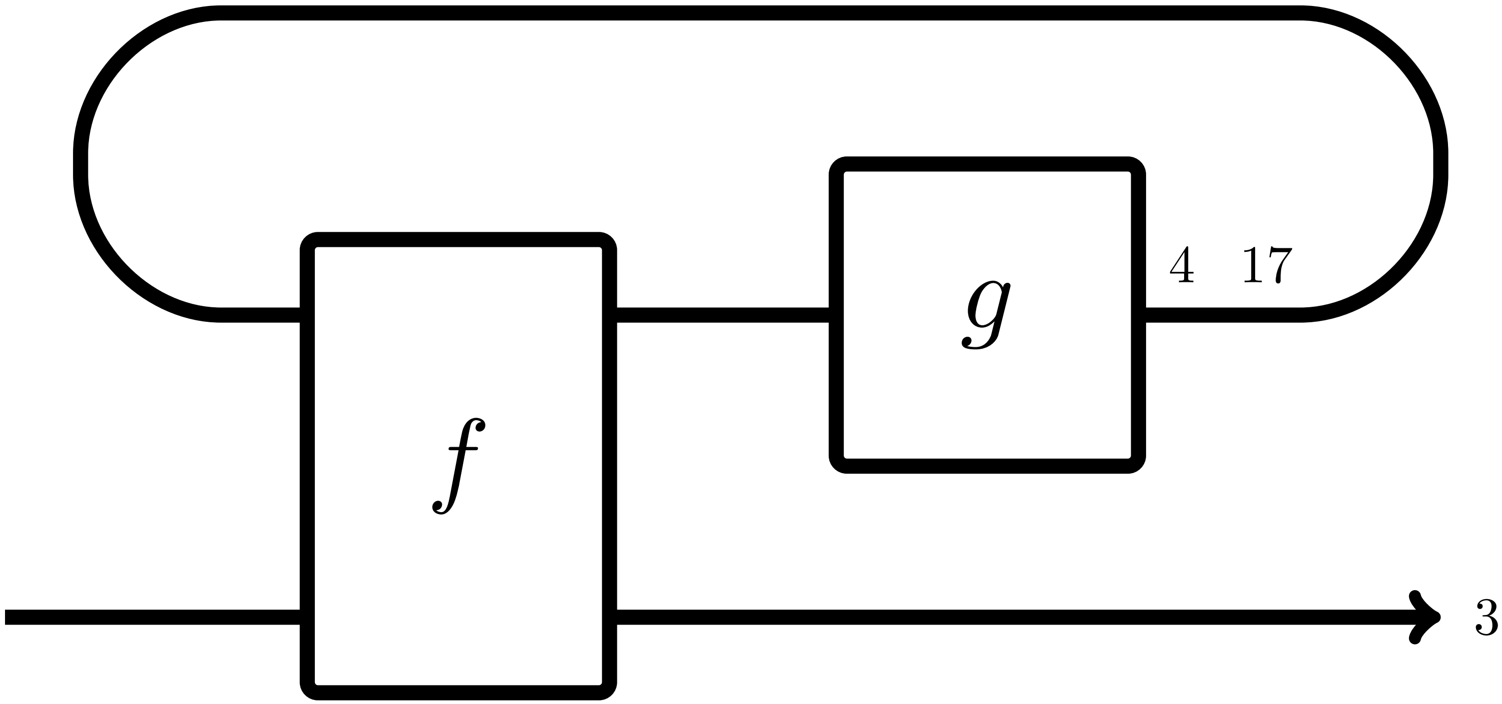
- f can remember interaction history while g can not
- f and g may behave nondeterministically

Interaction



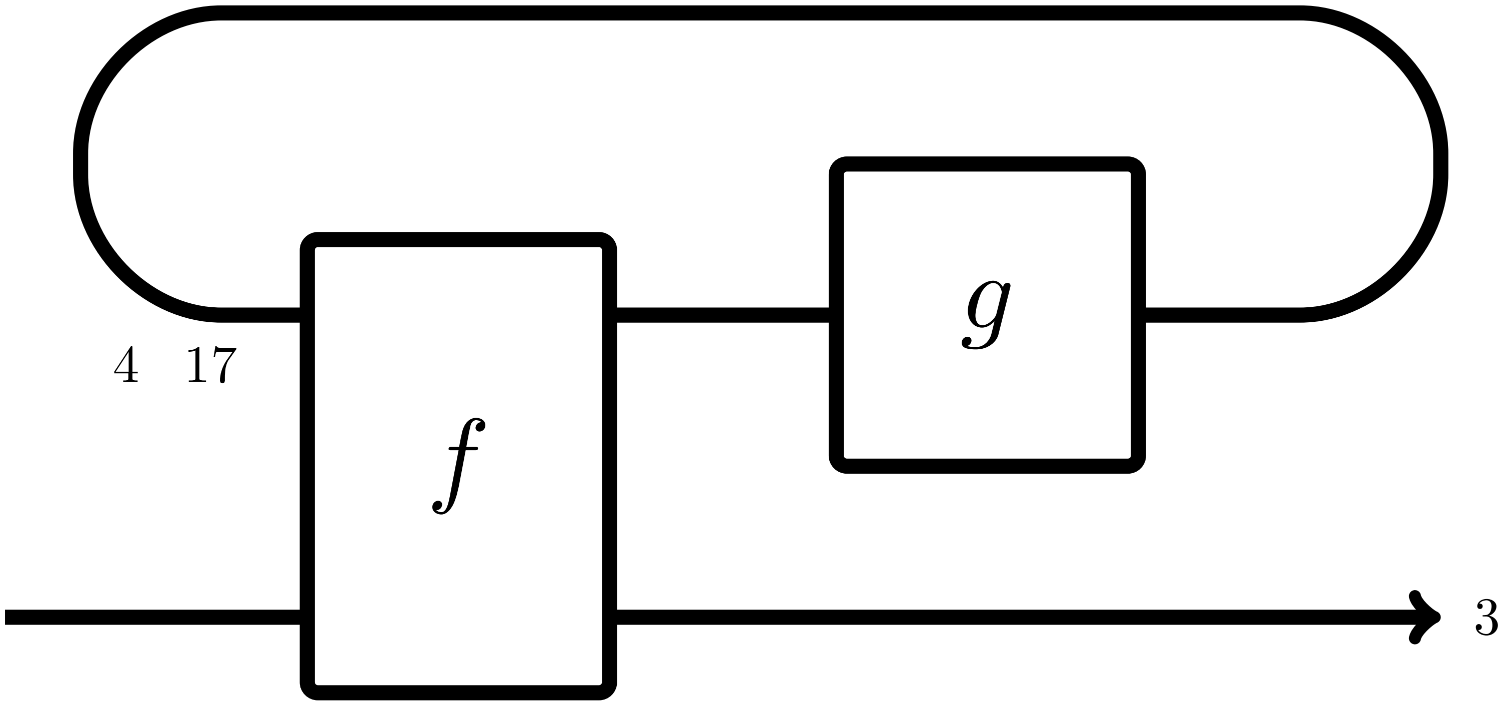
- f can remember interaction history while g can not
- f and g may behave nondeterministically

Interaction



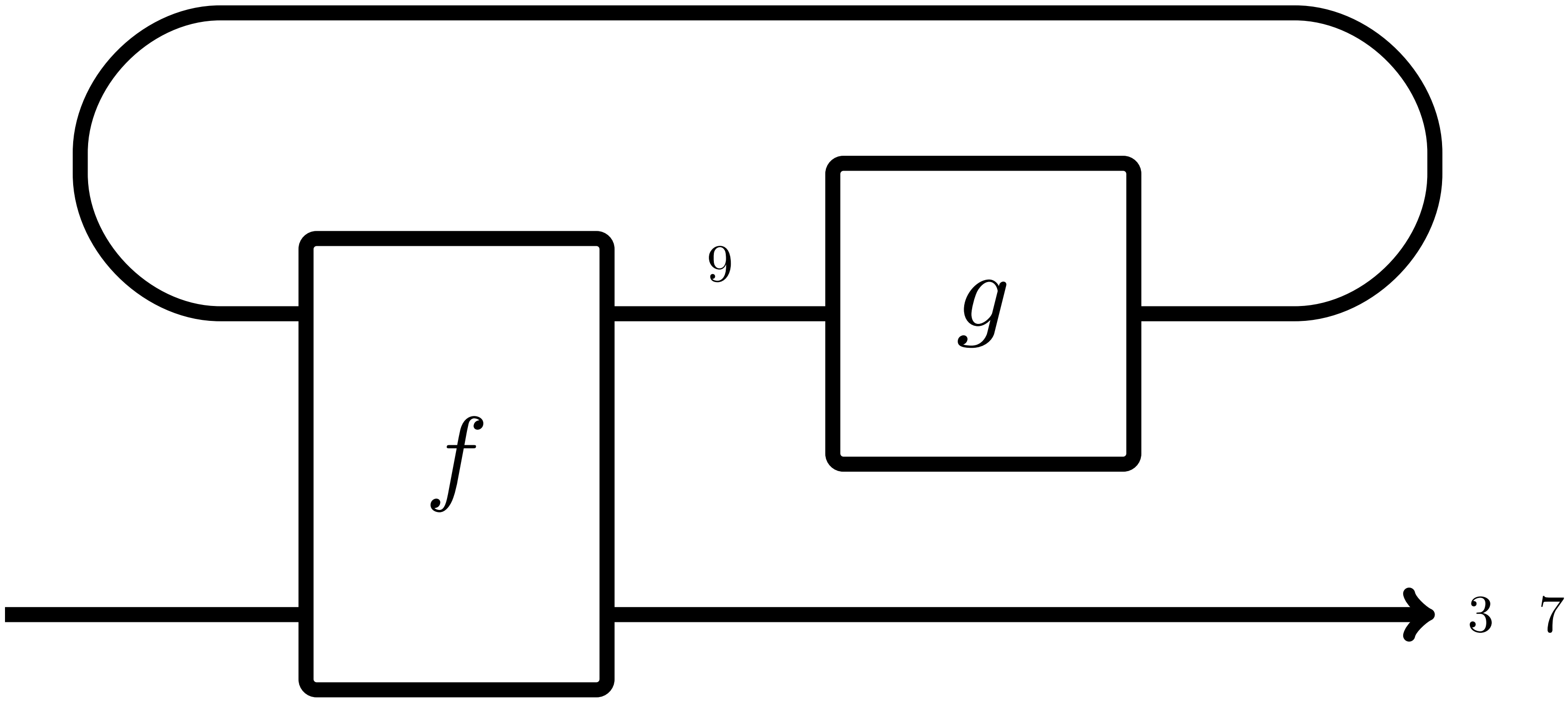
- f can remember interaction history while g can not
- f and g may behave nondeterministically

Interaction



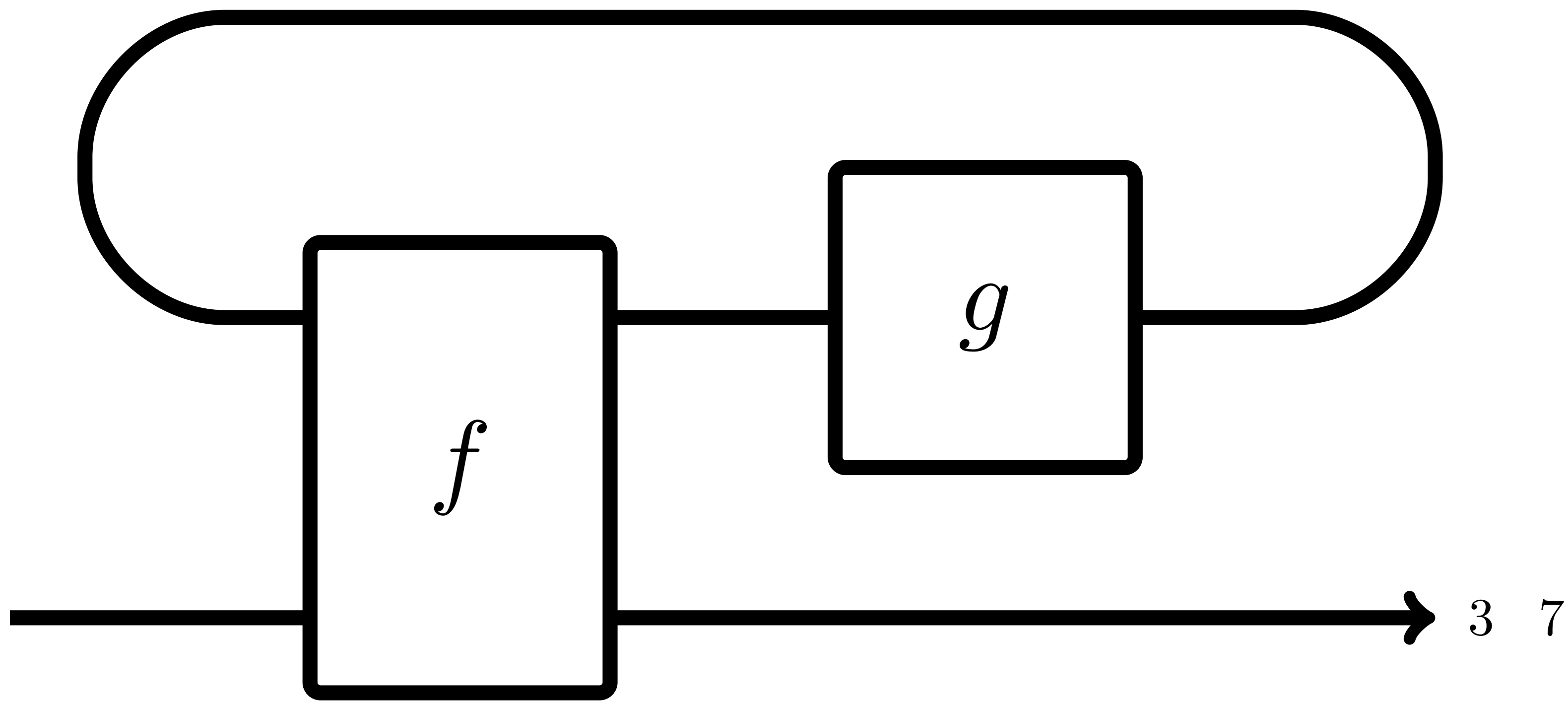
- f can remember interaction history while g can not
- f and g may behave nondeterministically

Interaction



- f can remember interaction history while g can not
- f and g may behave nondeterministically

Interaction



- f can remember interaction history while g can not
- f and g may behave nondeterministically

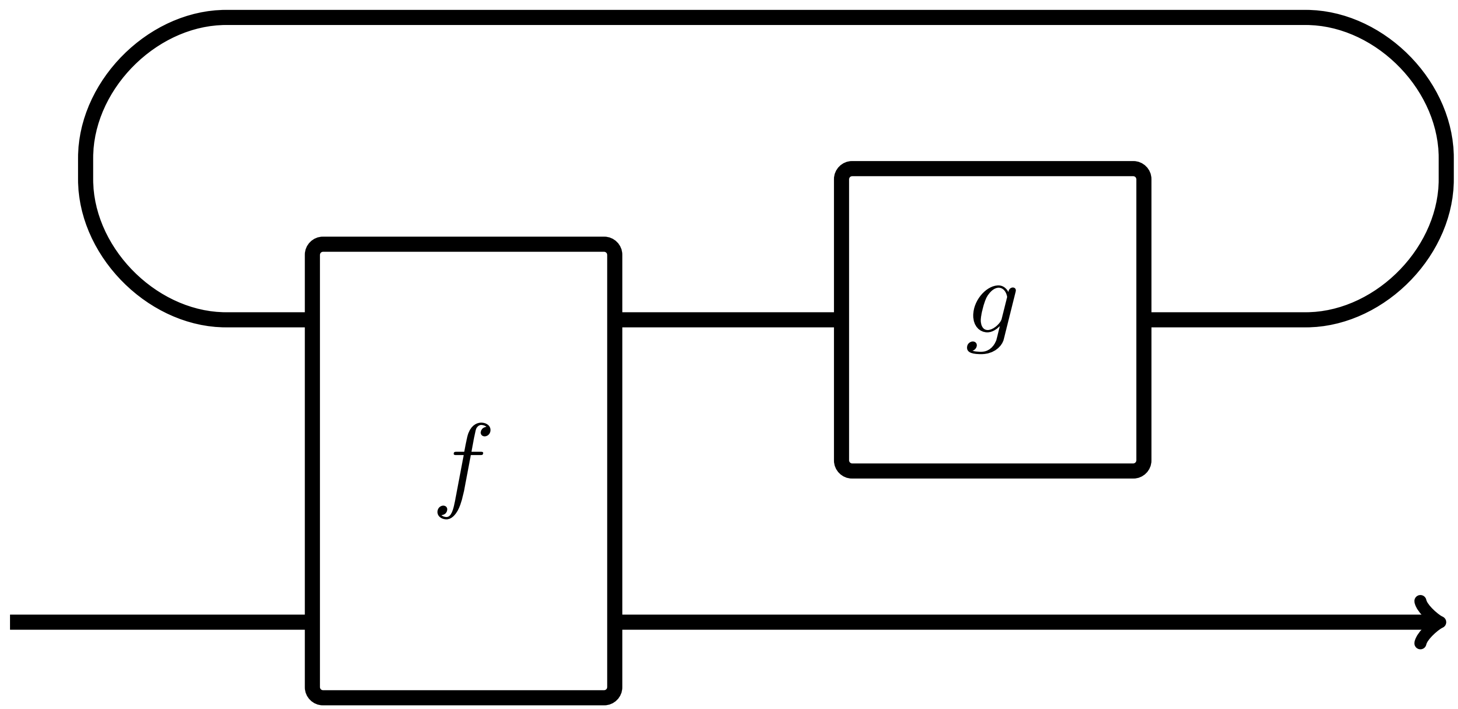
Weighted Relations

Def $f, g\colon \mathbf{N} \times \mathbf{N} \to \mathbf{N} \cup \{\infty\}$

$$
\begin{aligned}
(f \cdot g)(x,y) = \; & f(2x+1, 2y+1) + \\
& \sum_{\vec{x},\vec{y}} \quad
\begin{array}{l}
f(2x+1, 2x_1) \times g(x_1, y_1) \times \\
f([x, y_1], 2x_2) \times g(x_2, y_2) \times \\
f([x, y_1, y_2], 2x_3) \times g(x_3, y_3) \times \\
\quad \vdots \\
f([x, y_1, \ldots, y_n], 2y+1)
\end{array}
\end{aligned}
$$

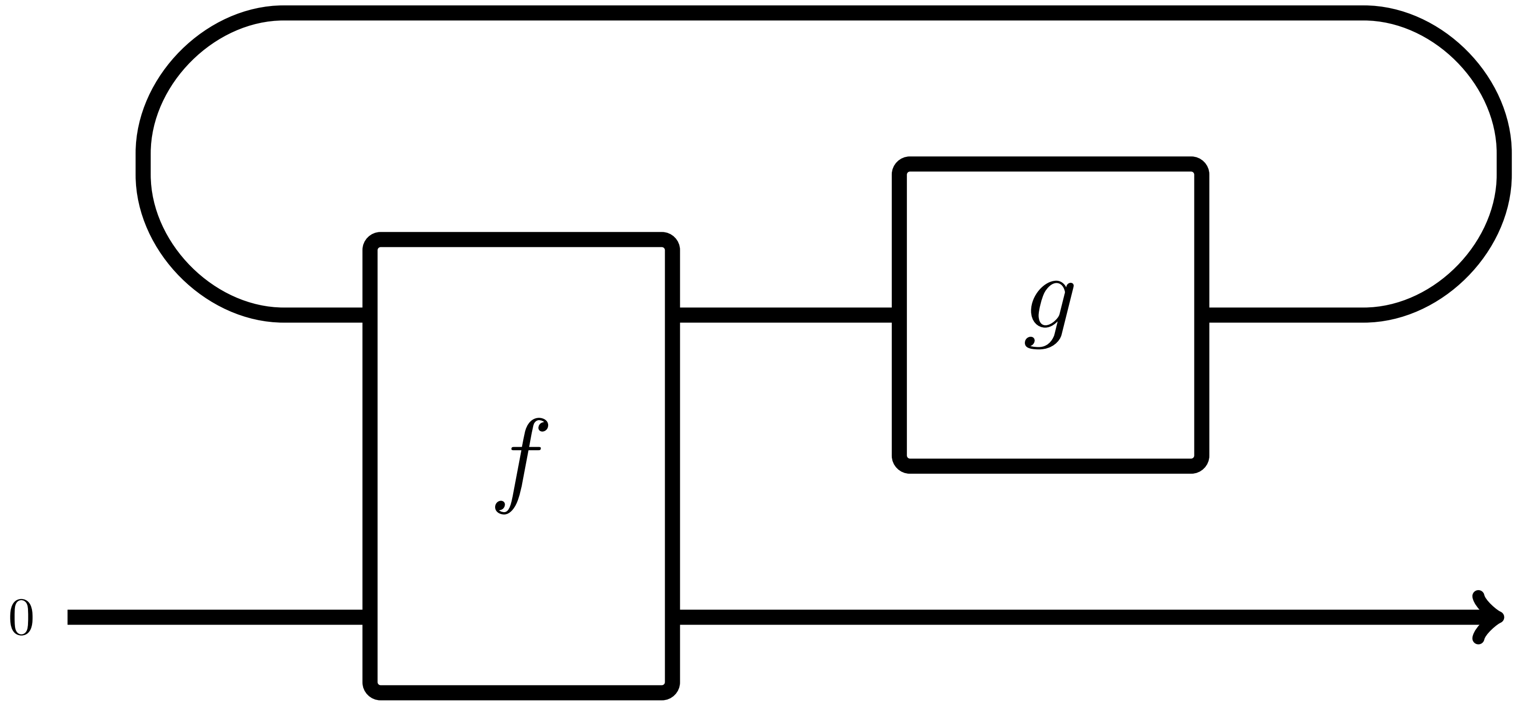
Prop. $\mathbf{WRel}(\mathbf{N}, \mathbf{N})$ is an SK-algebra.

Interaction



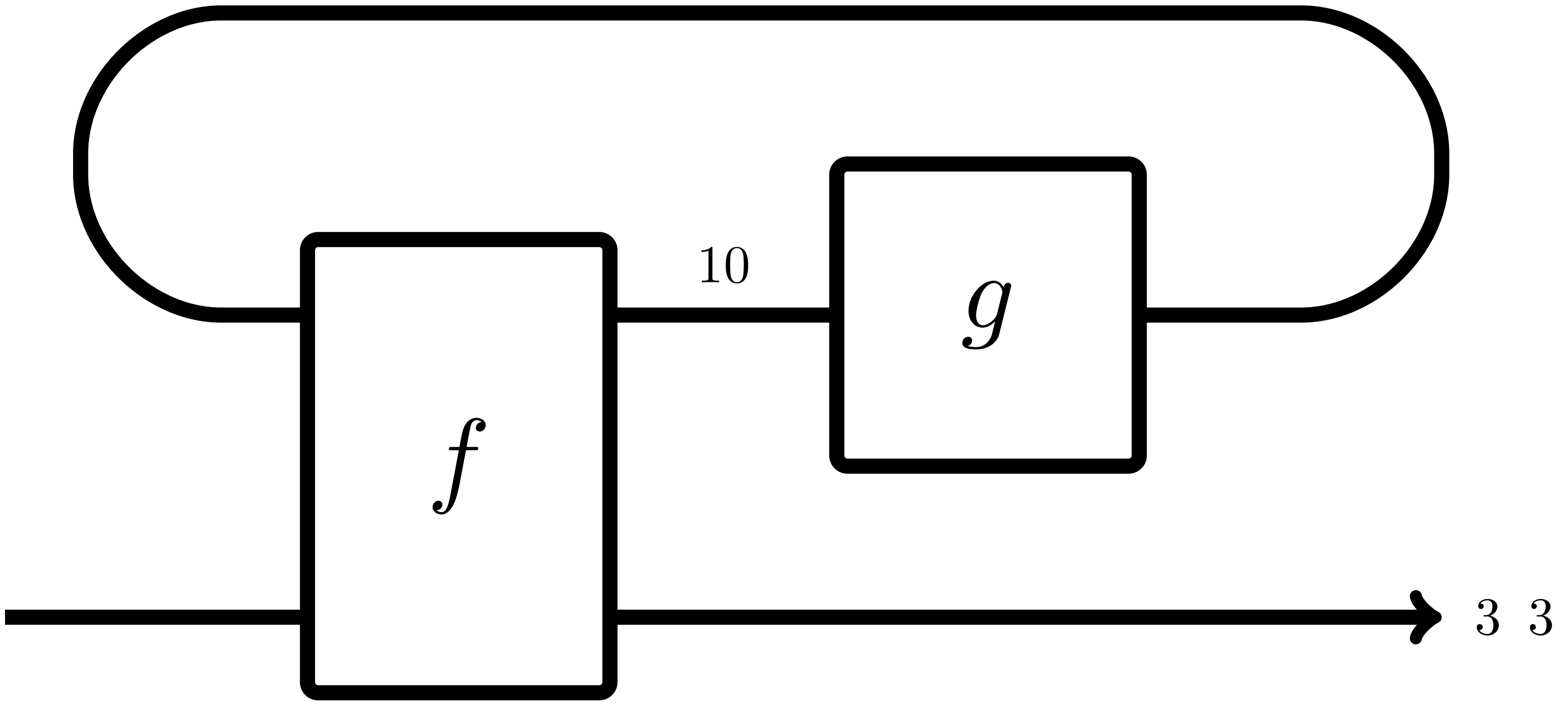
- f can remember interaction history while g can not
- f and g may behave nondeterministically
- $f(x, y) = 2 \iff f$ outputs two y

Interaction



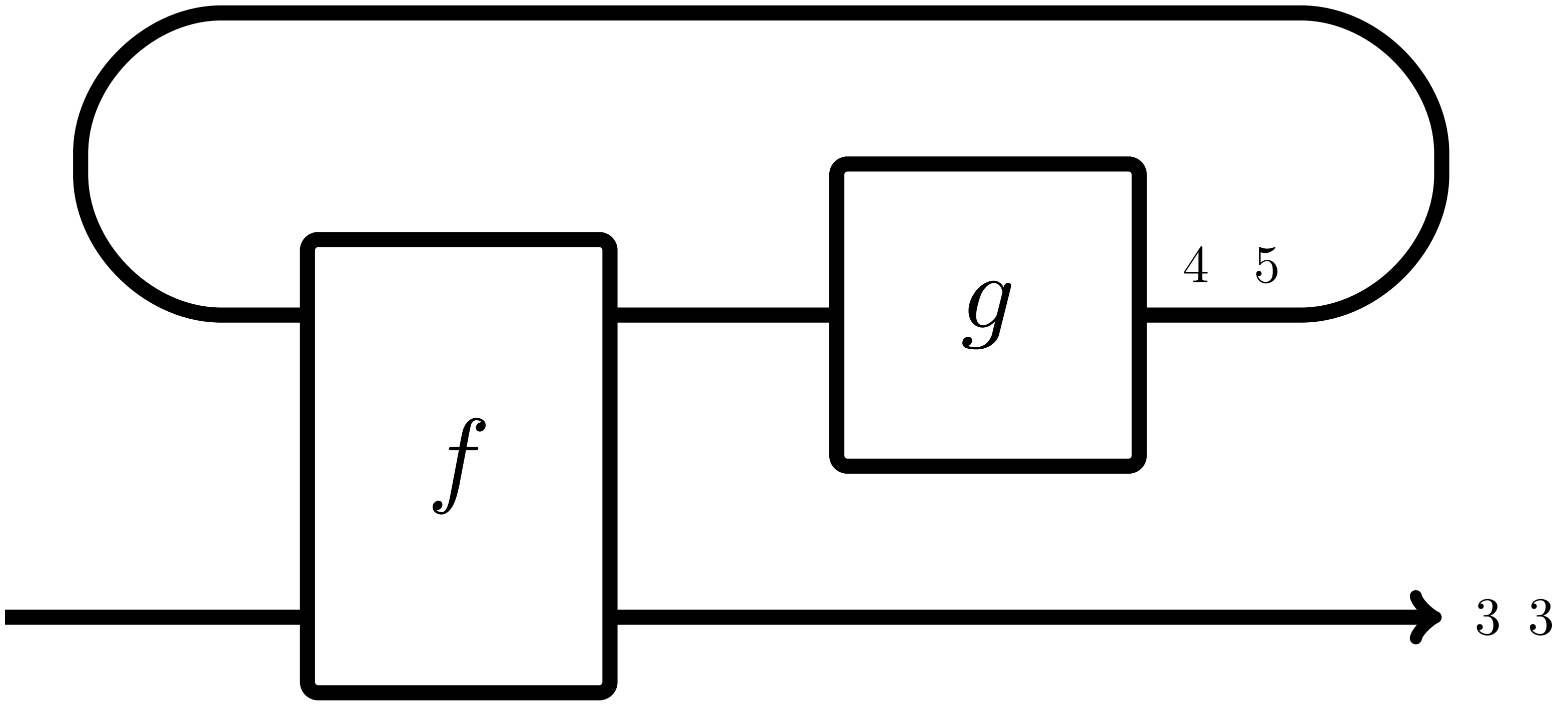
- f can remember interaction history while g can not
- f and g may behave nondeterministically
- $f(x, y) = 2 \iff f$ outputs two y

Interaction



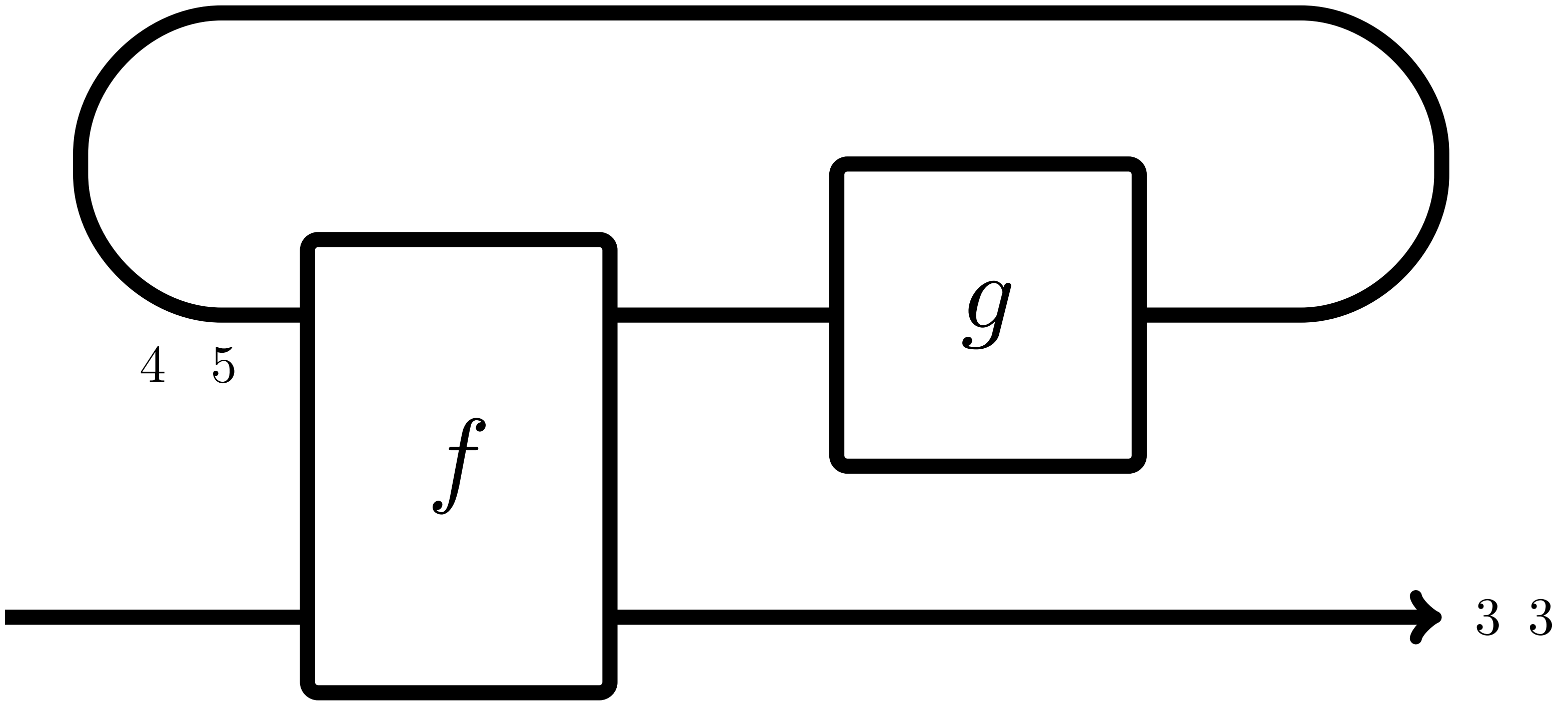
- f can remember interaction history while g can not
- f and g may behave nondeterministically
- $f(x, y) = 2 \iff f$ outputs two y

Interaction



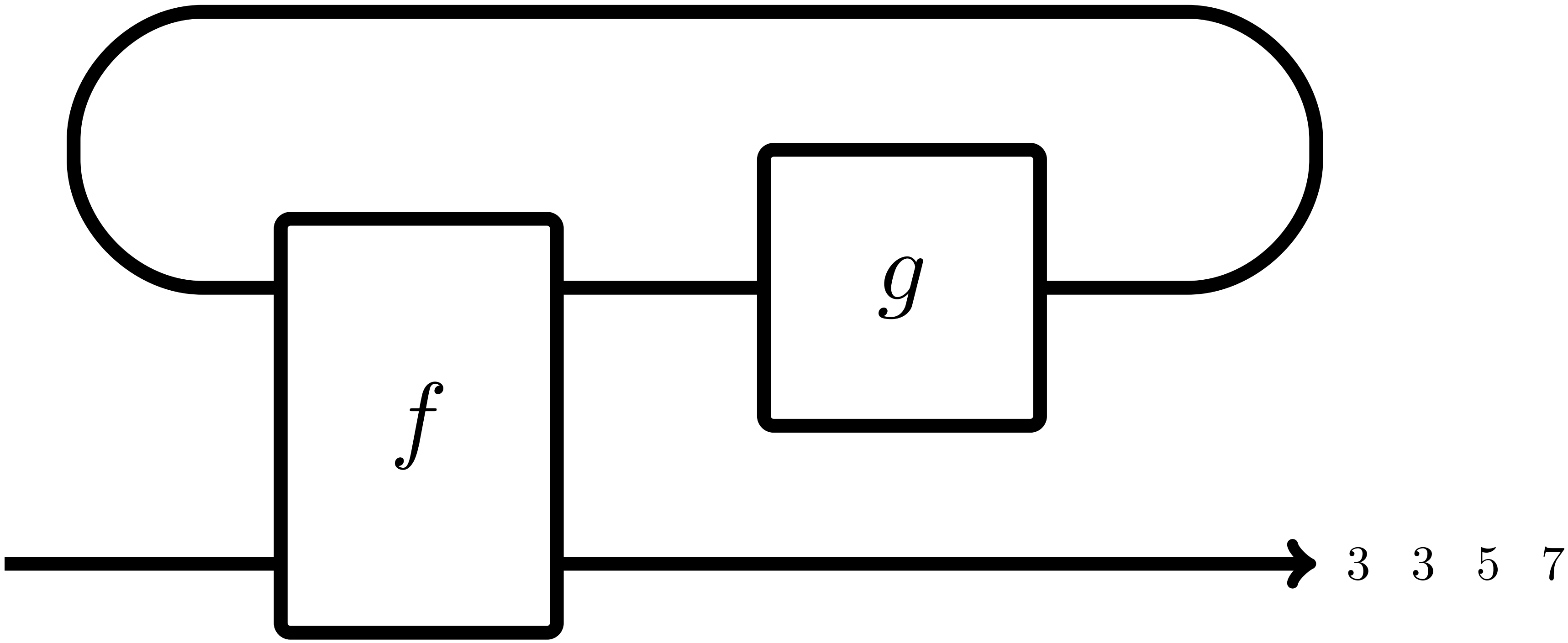
- f can remember interaction history while g can not
- f and g may behave nondeterministically
- $f(x, y) = 2 \iff f$ outputs two y

Interaction



- f can remember interaction history while g can not
- f and g may behave nondeterministically
- $f(x, y) = 2 \iff f$ outputs two y

Interaction



- f can remember interaction history while g can not
- f and g may behave nondeterministically
- $f(x, y) = 2 \iff f$ outputs two y

PER

Def. Let X and Y be partial equivalence relations on an SK-algebra A. A realizable function from X to Y is a function

$$f\colon A/X \to A/Y$$

such that there is a realizer $r \in A$ of f, i.e.,

$$f[a]_X = [ra]_Y$$

for any $[a]_X \in A/X$.

Prop $\mathbf{Per}(A)$ is a cartesian closed category with a natural number object

Main Results

Thm The full subCCC of $\mathbf{Cpo}$ generated by $\mathbf{N}_\perp$ is a full subcategory of $\mathbf{Per}(\mathbf{Rel}(\mathbf{N},\mathbf{N}))$

Thm The full subCCC of $\mathbf{Coh}$ generated by $\mathbf{N}_\perp$ is a full subcategory of $\mathbf{Per}(\mathbf{WRel}(\mathbf{N},\mathbf{N}))$

Thm [Oosten '99] The full subCCC of $\mathbf{HCoh}$ generated by $\mathbf{N}_\perp$ is a full subcategory of $\mathbf{Per}(\mathbf{Pfn}(\mathbf{N},\mathbf{N}))$

Remark

$\mathbf{Cpo}$ is a fully abstract model of PCF+$\mathbf{por}$+$\mathbf{exists}$ [Plotkin '77]

$\mathbf{Coh}$ is a fully abstract model of PCF+$\mathbf{strict}$+$\mathbf{gustave}$ [Paolini '06]

$\mathbf{Per}(\mathbf{Pfn}(\mathbf{N},\mathbf{N}))$ is a fully abstract model of PCF+$\mathbf{H}$ [Longley '02]

Outline of Proof

1. We define a category $\mathbf{rCoh}$ of coherence spaces "realized by $\mathbf{WRel}(\mathrm{N}, \mathrm{N})$"

2. Show that $U\colon \mathbf{rCoh} \to \mathbf{Coh}$ is an embedding

3. Show that $U'\colon \mathbf{rCoh} \to \mathbf{Per}(\mathbf{WRel}(\mathbf{N}, \mathbf{N}))$ is an embedding

4. Show that $\mathbf{rCoh}$ is cartesian closed and has $\mathbf{N}_\perp$

Notations:

$$
\begin{aligned}
A_{\mathbf{Rel}} &= \mathbf{Rel}(\mathbf{N}, \mathbf{N}) \\
A_{\mathbf{WRel}} &= \mathbf{WRel}(\mathbf{N}, \mathbf{N}) \\
A_{\mathbf{Pfn}} &= \mathbf{Pfn}(\mathbf{N}, \mathbf{N})
\end{aligned}
$$

Example

A program

$$\mathbf{pconv} : \mathbf{Nat} \Rightarrow \mathbf{Nat} \Rightarrow \mathbf{Nat}$$

such that

- if $M \longrightarrow^* 0$ then $\mathbf{pconv}\, M\, N \longrightarrow^* 0$
- if $N \longrightarrow^* 0$ then $\mathbf{pconv}\, M\, N \longrightarrow^* 0$
- otherwise, $\mathbf{pconv}\, M\, N \longrightarrow^\infty$

can be modeled in $\mathbf{Per}(A_{\mathbf{Rel}})$

can not be modeled in $\mathbf{Per}(A_{\mathbf{Pfn}})$

can not be modeled in $\mathbf{Per}(A_{\mathbf{WRel}})$

$\mathbf{pconv} \in \mathbf{Per}(A_{\mathbf{Rel}})$

SITUATION:

You want to drink beer. There are two people:

one exchanges a coin for a bottle of beer

one exchanges a coin for a bottle of beer or eats your coin

GOAL:

Get a bottle of beer

SOLUTION:

Try both

$\mathbf{pconv} \notin \mathbf{Per}(A_{\mathbf{Pfn}})$

SITUATION:

You want to drink beer. There are two people:

- one exchanges a coin for a bottle of beer
- one exchanges a coin for a bottle of beer or eat your coin

GOAL:

Get a bottle of beer

CONDITION:

You only have one coin

$\mathbf{pconv} \notin \mathbf{Per}(A_{\mathbf{WRel}})$

SITUATION:

You want to drink beer. There are two people:

- one exchanges a coin for a bottle of beer
- one exchanges a coin for a bottle of beer or eat your coin

GOAL:

Get a bottle of beer

CONDITION:

You should get exactly one bottle of beer

Example

A program

$$\mathbf{gustave} \colon \mathbf{Nat} \Rightarrow \mathbf{Nat} \Rightarrow \mathbf{Nat} \Rightarrow \mathbf{Nat}$$

such that

- if $M \longrightarrow^* 0$ and $N \longrightarrow^* 1$ then $\mathbf{gustave}\, M\ N\ L \longrightarrow^* 0$
- if $N \longrightarrow^* 0$ and $L \longrightarrow^* 1$ then $\mathbf{gustave}\, M\ N\ L \longrightarrow^* 1$
- if $L \longrightarrow^* 0$ and $M \longrightarrow^* 1$ then $\mathbf{gustave}\, M\ N\ L \longrightarrow^* 2$
- otherwise, $\mathbf{gustave}\, M\ N\ L \longrightarrow^\infty$

can be modeled in $\mathbf{Per}(A_{\mathbf{Rel}})$

can not be modeled in $\mathbf{Per}(A_{\mathbf{Pfn}})$

can be modeled in $\mathbf{Per}(A_{\mathbf{WRel}})$

$\mathbf{gustave} \in \mathbf{Per}(A_{\mathbf{Rel}})$

SITUATION:

You want to drink beer. There are three people:

Alice exchanges a coin for a ticket
Bob exchanges a ticket for a bottle of beer
Carol eats metal and paper

But you don't know who is Carol.

GOAL: Get a bottle of beer, and report who is Carol

SOLUTION: Try all patterns

$\mathbf{gustave} \in \mathbf{Per}(A_{\mathbf{WRel}})$

SITUATION:

You want to drink beer. There are three people:

Alice exchanges a coin for a ticket
Bob exchanges a ticket for a bottle of beer
Carol eats metal and paper

But you don't know who is Carol.

GOAL: Get a bottle of beer, and report who is Carol

CONDITION: You should get exactly one bottle of beer

SOLUTION: Try all patterns

$\mathbf{gustave} \notin \mathbf{Per}(A_{\mathbf{Pfn}})$

SITUATION:

You want to drink beer. There are three people:

Alice exchanges a coin for a ticket
Bob exchanges a ticket for a bottle of beer
Carol eats metal and paper

But you don't know who is Carol.

GOAL: Get a bottle of beer, and report who is Carol

CONDITION: You only have one coin

Example

A function

$$\mathbf{strict}\colon (\mathbf{Nat} \Rightarrow \mathbf{Nat}) \Rightarrow \mathbf{Nat}$$

such that

- if $u(\Omega) \longrightarrow^* 0$ then $\mathbf{strict}(u) \longrightarrow^* 0$
- if $u(\Omega) \longrightarrow^\infty$ and $u(0) \longrightarrow^* 0$ then $\mathbf{strict}(u) \longrightarrow^* 1$
- otherwise, $\mathbf{strict}(u) \longrightarrow^\infty$

where $\Omega\colon \mathbf{Nat} \Rightarrow \mathbf{Nat}$ is defined by $\Omega\ x = \Omega\ x$.

can be modeled in $\mathbf{Per}(A_{\mathbf{Rel}})$

can not be modeled in $\mathbf{Per}(A_{\mathbf{Pfn}})$

can be modeled in $\mathbf{Per}(A_{\mathbf{WRel}})$

$\mathbf{strict} \in \mathbf{Per}(A_{\mathbf{Pfn}})$

SITUATION: A vending machine behaves as follows

1: button $\rightarrow$ coin $\rightarrow$ a bottle of beer

2: button $\rightarrow$ a bottle of beer

Goal: Check the vending machine works correctly

ASSUMPTION: the vending machine is deterministic

SOLUTION: Try!

$\mathbf{strict} \notin \mathbf{Per}(A_{\mathbf{Rel}})$

SITUATION: A vending machine behaves as follows

1: button $\rightarrow$ coin $\rightarrow$ a bottle of beer

2: button $\rightarrow$ a bottle of beer

Goal: Check that the vending machine works correctly

ASSUMPTION: the vending machine is nondeterministic

$\mathbf{strict} \in \mathbf{Per}(A_{\mathbf{WRel}})$

SITUATION: A vending machine behaves as follows

1: button $\to$ coin $\to$ a bottle of beer

2: button $\to$ a bottle of beer

Goal: Check that the vending machine works correctly

ASSUMPTION1: the vending machine is nondeterministic

ASSUMPTION2: one bottle of beer for at most one coin

SOLUTION: Try!

There are many SK-algebra to be explored

- $\mathbf{N} \to (\mathbf{R}_+ \cup \{\infty\})^{\mathbf{N}}$
- $\{d \colon \mathbf{N} \to [0,1]^{\mathbf{N}} \mid \sum_n d(n) \leq 1\}$
- $\mathbf{Mealy}(\mathbf{N}, \mathbf{N})$
- $\mathbf{nMealy}(\mathbf{N}, \mathbf{N})$
- $\mathbf{pMealy}(\mathbf{N}, \mathbf{N})$

Thank you.