Trace Everywhere


Ichiro Hasuo
University of Tokyo (JP)
Three “Traces”

Coalgebraic $\text{Trace}$ Semantics

$\text{Traced}$ monoidal category

Quantum $\lambda$-calculus

Hasuo (Tokyo)
Three “Traces”

Coalgebraic Trace Semantics

Traced monoidal category

Quantum $\lambda$-calculus

Coinduction in $Kl(B)$

$FX \xrightarrow{F\text{beh}(c)} FZ$

$c \uparrow$

$X \xrightarrow{\text{beh}(c)} Y$

$\uparrow_{\text{final}}$

Tuesday, October 9, 12
Three “Traces”

Coalgebraic Trace Semantics

Traced monoidal category

Quantum \( \lambda \)-calculus

\[
\begin{align*}
F \text{beh}(c) & : FX \rightarrow FZ \\
c \uparrow & \quad \uparrow \text{final} \\
X \text{beh}(c) & \rightarrow Y
\end{align*}
\]

Coinduction in \( Kl(B) \)

Categorical GoI

[Abramsky, Haghverdi, Scott]
Three “Traces”

Coalgebraic \textbf{Trace} Semantics

\textbf{Traced} monoidal category

Quantum $\lambda$-calculus

Measurements by \textit{tracing out} matrices

Coinduction in $Kl(B)$

Categorical GoI

[Abramsky, Haghverdi, Scott]
Three “Traces”

Coalgebraic Trace Semantics

Traced monoidal category

Quantum λ-calculus

Goal: Denotational model of a quantum λ-calculus

Measurements by tracing out matrices

Categorical GoI

[Abramsky, Haghverdi, Scott]

Coinduction in $Kl(B)$
Three “Traces”

Coalgebraic Trace Semantics

Traced monoidal category

Quantum $\lambda$-calculus

appl

Goal: Denotational model of a quantum $\lambda$-calculus

Categorical GoI

[Abramsky, Haghverdi, Scott]

Hasuo (Tokyo)

\[
\begin{align*}
F_{\text{beh}(c)} : FX & \xrightarrow{\uparrow} FZ \\
X & \xrightarrow{\uparrow_{\text{final}}} Y
\end{align*}
\]

Coinduction in $KL(B)$
GoI: Geometry of Interaction

* J.-Y. Girard, at Logic Colloquium ’88
GoI: Geometry of Interaction

* J.-Y. Girard, at Logic Colloquium ’88
* Provides denotational semantics $\left[M\right]$ for linear $\lambda$-term $M$
GoI: Geometry of Interaction

* J.-Y. Girard, at Logic Colloquium ’88
* Provides denotational semantics $\left[{M}\right]$ for linear \(\lambda\)-term \(M\)
* In this talk:
  * Its categorical formulation
    [Abramsky, Haghverdi, Scott ’02]
  * “The GoI Animation”
The GoI Animation

\[ [M] = (\mathbb{N} \rightarrow \mathbb{N}, \text{ a partial function}) \]

= "piping"

\[ \begin{array}{cccccc}
& & & & & \\
& & & & & \\
& & & & & \\
& & & & & \\
1 & 2 & 3 & 4 & \cdots & \text{(countably many)}
\end{array} \]
The GoI Animation

\[
[M] = ( \mathbb{N} \rightarrow \mathbb{N}, \text{ a partial function } )
\]

= “piping”

1 2 3 4 ... (countably many)
\[
[M] = (\mathbb{N} \rightarrow \mathbb{N}, \text{ a partial function})
\]

= “piping”

\[
\begin{array}{cccc}
1 & 2 & 3 & 4 \\
\downarrow & \downarrow & \downarrow & \downarrow \\
\end{array}
\]

... (countably many)
The GoI Animation

\[
[M] = (\mathbb{N} \rightarrow \mathbb{N}, \text{a partial function})
\]

\[= \text{“piping”}
\]

... (countably many)

Hasuo (Tokyo)
The GoI Animation

\[
[M] = (\mathbb{N} \rightarrow \mathbb{N}, \text{a partial function})
\]

= “piping”

\[
\begin{align*}
1 & \quad \downarrow & \quad 2 & \quad \downarrow & \quad 3 & \quad \downarrow & \quad 4 & \quad \downarrow & \quad \ldots
\end{align*}
\]

(countably many)
\[ [M] = (\mathbb{N} \rightarrow \mathbb{N}, \text{a partial function}) \]

= “piping”

\[ \begin{array}{cccc}
1 & 2 & 3 & 4 \\
\downarrow & \downarrow & \downarrow & \downarrow \\
\end{array} \]

... (countably many)
\[ [M] = (\mathbb{N} \rightarrow \mathbb{N}, \text{ a partial function}) = \text{“piping”} \]

Hasuo (Tokyo)
The GoI Animation

* Function application $[MN]$

* by “parallel composition + hiding”
\[ M N \] = [M] ... [N]
\[ MN \]

= 

\[ M \]

\[ N \]
\[ M N \]

\[
\begin{bmatrix}
M \\
N
\end{bmatrix}
\]

\[
\begin{bmatrix}
M \\
N
\end{bmatrix} =
\]

\[
\begin{bmatrix}
M \\
N
\end{bmatrix}
\]
\[ MN \] = [M] [N]
\[ MN \]

= 

\[ M \]

\[ N \]
\[ MN \] = \[ M \] \parallel \[ N \] + hiding

"parallel composition + hiding" (cf. AJM games)
\[ \[M N\] = \]

\[ \[M\] \]

\[ \[N\] \]

\[
\begin{align*}
M &= \lambda x. x + 1 & N &= 2 \\
M &= \lambda x. 1 & N &= 2 \\
M &= \lambda f. f1 & N &= \lambda x. (x + 1)
\end{align*}
\]
\[ \boxed{\boxed{M N}} \]

\[ = \]

\[ \boxed{\boxed{M}} \quad \boxed{\boxed{N}} \]

\[ \rightarrow \quad M = \lambda x. \, x + 1 \quad N = 2 \]
\[ M = \lambda x. \, 1 \quad N = 2 \]
\[ M = \lambda f. \, f\!1 \quad N = \lambda x. \, (x + 1) \]
... $M = \lambda x. x + 1 \quad N = 2$

$M = \lambda x. 1 \quad N = 2$

$M = \lambda f. f1 \quad N = \lambda x. (x + 1)$
\[ MN \] =

\[ M = \lambda x. x + 1 \quad N = 2 \]
\[ M = \lambda x. 1 \quad N = 2 \]
\[ M = \lambda f. f1 \quad N = \lambda x. (x + 1) \]
\[ M = \lambda x. x + 1 \quad N = 2 \]
\[ M = \lambda x. I \quad N = I + x + 1 \]
\[ [MN] \]

... 

\[ [M] \]  
\[ [N] \] 

... 

\[ M = \lambda x. x + 1 \]  
\[ N = 2 \] 

\[ M = \lambda x. 1 \]  
\[ N = 2 \] 

\[ M = \lambda f. f1 \]  
\[ N = \lambda x. (x + 1) \]
\[ M N \]

\[ = \]

\[ M = \lambda x. x + 1 \quad N = 2 \]

\[ M = \lambda x. 1 \quad N = 2 \]

\[ M = \lambda f. f1 \quad N = \lambda x. (x + 1) \]
\[ [MN] = [M] \odot [N] \]

\[
\begin{align*}
M &= \lambda x. x + 1 \\
N &= 2
\end{align*}
\]

\[
\begin{align*}
M &= \lambda x. 1 \\
N &= 2
\end{align*}
\]

\[
\begin{align*}
M &= \lambda f. f \, 1 \\
N &= \lambda x. (x + 1)
\end{align*}
\]
$[MN] = \ldots$

$M = \lambda x. x + 1$

$N = 2$

$M = \lambda x. 1$

$N = 2$

$M = \lambda f. f1$

$N = \lambda x. (x + 1)$
\[ [MN] \]

\[ = \]

\[ \cdots \]

\[ [M] \]

\[ [N] \]

\[ [MN] \]

\[ M = \lambda x. x + 1 \quad N = 2 \]

\[ M = \lambda x. 1 \quad N = 2 \]

\[ \rightarrow M = \lambda f. f1 \quad N = \lambda x. (x + 1) \]
\[MN\] = [...]

\[M\] = \(\lambda x. x + 1\)
\[N\] = 2

\[M\] = \(\lambda x. 1\)
\[N\] = 2

\[M\] = \(\lambda f. f 1\)
\[N\] = \(\lambda x. (x + 1)\)
\[MN\] = [...]

\[M\] = \(\lambda x. x + 1\)  \(N = 2\)

\[N\] = \(\lambda x. 1\)  \(N = 2\)

\[M\] = \(\lambda f. f1\)  \(N = \lambda x. (x + 1)\)
\[
\begin{align*}
M N & = \lambda x. x + 1 & N & = 2 \\
M & = \lambda x. 1 & N & = 2 \\
\rightarrow M & = \lambda f. f1 & N & = \lambda x. (x + 1)
\end{align*}
\]
\[MN\] = \[M\] \[N\]

\[M = \lambda x. x + 1 \quad N = 2\]
\[M = \lambda x. 1 \quad N = 2\]
\[M = \lambda f. f1 \quad N = \lambda x. (x + 1)\]
GoI: Geometry of Interaction

* J.-Y. Girard, at Logic Colloquium ’88
GoI: Geometry of Interaction

* J.-Y. Girard, at Logic Colloquium ’88
* Provides denotational semantics $[M]$ for linear $\lambda$-term $M$
GoI: Geometry of Interaction

* J.-Y. Girard, at Logic Colloquium ’88
* Provides denotational semantics $[M]$ for linear $\lambda$-term $M$
* Similar to game semantics [AJM/HO]
GoI: Geometry of Interaction

* J.-Y. Girard, at Logic Colloquium ’88
* Provides denotational semantics $[M]$ for linear $\lambda$-term $M$
  
  * Similar to game semantics $[\text{AJM/HO}]$
  
  * Linearity: simplicity; no-cloning
GoI: Geometry of Interaction

* J.-Y. Girard, at Logic Colloquium ’88
* Provides denotational semantics $[M]$ for linear $\lambda$-term $M$
* Similar to game semantics $[AJM/HO]$
* Linearity: simplicity; no-cloning
* Girard translation $A \rightarrow B$ as $!A \rightarrow B$
GoI: Geometry of Interaction

- J.-Y. Girard, at Logic Colloquium ’88
- Provides denotational semantics $\llbracket M \rrbracket$ for linear $\lambda$-term $M$
- Similar to game semantics [AJM/HO]
- Linearity: simplicity; no-cloning
- Girard translation
- “Geometry”: invariant under $\beta$-reductions

$$\begin{array}{c}
A \rightarrow B \\
as \quad !A \rightarrow B
\end{array}$$

$$\boxempty = |$$

(Tokyo)
Categorical GoI

* Axiomatics of GoI in the categorical language

* Our main reference:
  

  * Especially its technical report version (Oxford CL), since it’s a bit more detailed
The Categorical GoI Workflow

- Traced monoidal category $\mathcal{C}$
  + other constructs $\rightarrow$ "GoI situation" [AHS02]

- Categorical GoI [AHS02]

- Linear combinatory algebra

- Realizability

- Linear category
The Categorical GoI

Workflow

Traced monoidal category $\mathcal{C}$
+ other constructs $\rightarrow$ "GoI situation" [AHS02]

Categorical GoI [AHS02]

Linear combinatory algebra

Realizability

Linear category
The Categorical GoI Workflow

Traced monoidal category $C$
+ other constructs $\rightarrow$ “GoI situation” [AHS02]

Categorical GoI [AHS02]

Linear combinatory algebra

Realizability

Linear category

* Applicative str. + combinators
* Model of untyped calculus

Hasuo (Tokyo)
The Categorical GoI Workflow

- Traced monoidal category $C$
  + other constructs $\rightarrow$ “GoI situation” [AHS02]

- Categorical GoI [AHS02]
  - Applicative str. + combinators
  - Model of untyped calculus

- Linear combinatory algebra

- Realizability

- Linear category
  - Model of typed calculus

Hasuo (Tokyo)
The Categorical GoI Workflow

Traced monoidal category $\mathcal{C}$ + other constructs $\rightarrow$ “GoI situation” [AHS02]

Categorical GoI [AHS02]

Linear combinatory algebra

Realizability

Linear category

Applicative str. + combinators

Model of untyped calculus

PER, $\omega$-set, assembly, ...

“Programming in untyped $\lambda$”

Model of typed calculus
The Categorical GoI Workflow

Traced monoidal category $\mathcal{C}$
+ other constructs $\rightarrow$ “GoI situation” [AHS02]

Categorical GoI [AHS02]

Linear combinatory algebra

Realizability

Linear category

Model of typed calculus

Applicative str. + combinators

Model of untyped calculus

PER, $\omega$-set, assembly, ...

“Programming in untyped $\lambda$”
Defn. (LCA)
A linear combinatory algebra (LCA) is a set $A$ equipped with

- a binary operator (called an *applicative structure*)
  
  $\cdot : A^2 \rightarrow A$

- a unary operator
  
  $!: A \rightarrow A$

- (combinators) distinguished elements $B, C, I, K, W, D, \delta, F$
satisfying

\[
\begin{align*}
Bxyz &= x(yz) & \text{Composition, Cut} \\
Cxyz &= (xz)y & \text{Exchange} \\
I &= x & \text{Identity} \\
K &= x & \text{Weakening} \\
W &= x & \text{Contraction} \\
D &= x & \text{Dereliction} \\
\delta &= x & \text{Comultiplication} \\
F &= xy & \text{Monoidal functoriality}
\end{align*}
\]

Here: $\cdot$ associates to the left; $\cdot$ is suppressed; and $!$ binds stronger than $\cdot$ does.
**Defn. (LCA)**

A linear combinatory algebra (LCA) is a set $A$ equipped with

- a binary operator (called an *applicative structure*)

  $$ \cdot : A^2 \rightarrow A $$

- a unary operator

  $$ ! : A \rightarrow A $$

- *(combinators)* distinguished elements $B, C, I, K, W, D, \delta, F$ satisfying

  
  \[
  \begin{align*}
  Bxyz &= x(yz) & \text{Composition, Cut} \\
  Cxyz &= (xz)y & \text{Exchange} \\
  lx &= x & \text{Identity} \\
  Kxy &= x & \text{Weakening} \\
  Wxy &= x!!y & \text{Contraction} \\
  Dx &= x & \text{Dereliction} \\
  \delta x &= !!!x & \text{Comultiplication} \\
  Fxy &= !(!x) & \text{Monoidal functoriality}
  \end{align*}
  \]

Here: $\cdot$ associates to the left; $\cdot$ is suppressed; and $!$ binds stronger than $\cdot$ does.
Defn. (LCA)
A linear combinatory algebra (LCA) is a set $A$ equipped with

- a binary operator (called an applicative structure)
  \[ \cdot : A^2 \rightarrow A \]
- a unary operator
  \[ ! : A \rightarrow A \]
- (combinators) distinguished elements $B, C, I, K, W, D, \delta, F$
  satisfying

\begin{align*}
Bxyz &= x(yz) & \text{Composition, Cut} \\
Cxyz &= (xz)y & \text{Exchange} \\
Ix &= x & \text{Identity} \\
Kx!y &= x & \text{Weakening} \\
Wx!y &= x!y!y & \text{Contraction} \\
D!x &= x & \text{Dereliction} \\
\delta!x &= !!x & \text{Comultiplication} \\
F!x!y &= !(xy) & \text{Monoidal functoriality}
\end{align*}

Here: $\cdot$ associates to the left; $\cdot$ is suppressed; and $!$ binds stronger than $\cdot$ does.
**Linear Combinatory Algebra (LCA)**

**Defn. (LCA)**

A linear combinatory algebra (LCA) is a set $A$ equipped with

- a binary operator (called an applicative structure)
  
  $· : A^2 \rightarrow A$

- a unary operator
  
  $!: A \rightarrow A$

- (combinators) distinguished elements $B, C, I, K, W, D, δ, F$ satisfying

  $Bxyz = x(yz)$  
  Composition, Cut
  
  $Cxyz = (xz)y$  
  Exchange
  
  $lx = x$  
  Identity
  
  $Kx!y = x$  
  Weakening
  
  $Wx!y = x!y!y$  
  Contraction
  
  $Dx = x$  
  Dereliction
  
  $δx = !x$  
  Comultiplication
  
  $Fx!y = !(xy)$  
  Monoidal functoriality

Here: $·$ associates to the left; $·$ is suppressed; and $!$ binds stronger than $·$ does.

---

**What we want (outcome):**

- **Model of untyped linear $λ$**
- $a \in A \approx$ closed linear $λ$-term
Defn. (LCA)

A linear combinatory algebra (LCA) is a set $A$ equipped with

- a binary operator (called an applicative structure)
  $$\cdot: A^2 \to A$$

- a unary operator
  $$! : A \to A$$

- (combinators) distinguished elements $B, C, I, K, W, D, \delta, F$ satisfying

  $\begin{align*}
  Bxyz &= x(yz) & \text{Composition, Cut} \\
  Cxyz &= (xz)y & \text{Exchange} \\
  lx &= x & \text{Identity} \\
  Kxy &= x & \text{Weakening} \\
  Wxy &= x!y!y & \text{Contraction} \\
  Dx &= x & \text{Dereliction} \\
  \delta x &= !!x & \text{Comultiplication} \\
  Fxy &= !(xy) & \text{Monoidal functoriality}
  \end{align*}$

Here: $\cdot$ associates to the left; $\cdot$ is suppressed; and $!$ binds stronger than $\cdot$ does.
GoI situation

**Defn.** (GoI situation [AHS02])

A *GoI situation* is a triple \((C, F, U)\) where

- **\(C = (\mathbb{C}, \otimes, I)\)** is a traced symmetric monoidal category (TSMC);
- **\(F : C \to C\)** is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).
  
  \[
  \begin{align*}
  e : FF \triangleleft F & : e' \quad \text{Comultiplication} \\
  d : \text{id} \triangleleft F & : d' \quad \text{Dereliction} \\
  c : F \otimes F \triangleleft F & : c' \quad \text{Contraction} \\
  w : K_I \triangleleft F & : w' \quad \text{Weakening}
  \end{align*}
  \]

  Here \(K_I\) is the constant functor into the monoidal unit \(I\);

- **\(U \in C\)** is an object (called *reflexive object*), equipped with the following retractions.
  
  \[
  \begin{align*}
  j : U \otimes U \triangleleft U & : k \\
  & I \triangleleft U \\
  u : FU \triangleleft U & : v
  \end{align*}
  \]
**GoI situation**

**Defn.** (GoI situation [AHS02])

A GoI situation is a triple \((C, F, U)\) where

- \(C = (C, \otimes, I)\) is a traced symmetric monoidal category (TSMC);
- \(F : C \to C\) is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).
  - \(e : FF \triangleleft F : e'\)  Comultiplication
  - \(d : \text{id} \triangleleft F : d'\)  Dereliction
  - \(c : F \otimes F \triangleleft F : c'\)  Contraction
  - \(w : K_I \triangleleft F : w'\)  Weakening

Here \(K_I\) is the constant functor into the monoidal unit \(I\);

- \(U \in C\) is an object (called reflexive object), equipped with the following rejections.
  - \(j : U \otimes U \triangleleft U : k\)
  - \(I \triangleleft U\)
  - \(u : FU \triangleleft U : v\)

**Monoidal category \((C, \otimes, I)\)**

**String diagrams**
**GoI situation**

* Monoidal category \((\mathcal{C}, \otimes, I)\)

* String diagrams

---

**Defn.** (GoI situation [AHS02])

A *GoI situation* is a triple \((\mathcal{C}, F, U)\) where

- \(\mathcal{C} = (\mathcal{C}, \otimes, I)\) is a traced symmetric monoidal category (TSMC);
- \(F : \mathcal{C} \to \mathcal{C}\) is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).
  - \(\epsilon : FF \triangleleft F : \epsilon'\) Comultiplication
  - \(d : \text{id} \triangleleft F : d'\) Dereliction
  - \(c : F \otimes F \triangleleft F : c'\) Contraction
  - \(w : K_I \triangleleft F : w'\) Weakening

Here \(K_I\) is the constant functor into the monoidal unit \(I\);

- \(U \in \mathcal{C}\) is an object (called reflexive object), equipped with the following retractions.
  - \(j : U \otimes U \triangleleft U : k\)
  - \(I \triangleleft U\)
  - \(u : FU \triangleleft U : v\)
**GoI situation**

Defn. (GoI situation [AHS02])

A GoI situation is a triple $(C, F, U)$ where

- $C = (C, \otimes, I)$ is a traced symmetric monoidal category (TSMC);
- $F : C \to C$ is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).
  - $e : FF \otimes F \to F : e'$ (Comultiplication)
  - $d : \text{id} \otimes F \to F : d'$ (Dereliction)
  - $c : F \otimes F \otimes F \to F : c'$ (Contraction)
  - $w : K_I \otimes F \to F : w'$ (Weakening)

Here $K_I$ is the constant functor into the monoidal unit $I$;

- $U \in C$ is an object (called reflexive object), equipped with the following retractions.
  - $j : U \otimes U \otimes U \to U : k$
  - $I \otimes U$
  - $u : FU \otimes U \to U : v$

**Monoidal category** $(C, \otimes, I)$

**String diagrams**

\[
\begin{align*}
  &\quad A \xrightarrow{f} B \quad B \xrightarrow{g} C \\
  &\quad A \xrightarrow{g \circ f} C
\end{align*}
\]

\[
\begin{align*}
  &\quad A \xrightarrow{f} B \quad C \xrightarrow{g} D \\
  &\quad A \otimes C \xrightarrow{f \otimes g} B \otimes D \\
  &\quad h \circ (f \otimes g)
\end{align*}
\]
**GoI situation**

*Traced monoidal category*

* "feedback"

\[
\begin{align*}
A \otimes C & \overset{f}{\longrightarrow} B \otimes C \\
A & \overset{\text{tr}(f)}{\longrightarrow} B
\end{align*}
\]

that is

\[
\begin{tikzpicture}
\node (A) at (0,0) {$A$};
\node (B) at (0,-1) {$B$};
\node (C) at (1,0) {$C$};
\node (C') at (2,0) {$C$};
\node (A') at (2,0) {$A$};
\node (A'') at (2,-1) {$B$};
\draw[->] (A) -- (B) node[midway,above] {$f$};
\draw[->] (A') -- (A'') node[midway,above] {$\text{tr}(f)$};
\end{tikzpicture}
\]

**Defn. (GoI situation [AHS02])**

A GoI situation is a triple \((\mathcal{C}, F, U)\) where

- \(\mathcal{C} = (\mathcal{C}, \otimes, I)\) is a traced symmetric monoidal category (TSMC);
- \(F : \mathcal{C} \to \mathcal{C}\) is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).
  - \(e : FF \triangleleft F \to e'\) Comultiplication
  - \(d : \text{id} \triangleleft F \to d'\) Dereliction
  - \(c : F \otimes F \triangleleft F \to c'\) Contraction
  - \(w : K_I \triangleleft F \to w'\) Weakening

Here \(K_I\) is the constant functor into the monoidal unit \(I\);

- \(U \in \mathcal{C}\) is an object (called reflexive object), equipped with the following retractions.
  - \(j : U \otimes U \triangleleft U \to k\)
  - \(I \triangleleft U\)
  - \(u : FU \triangleleft U \to v\)
I use two ways of depicting partial functions $\mathbb{N} \rightarrow \mathbb{N}$.
I use two ways of depicting partial functions $\mathbb{N} \rightarrow \mathbb{N}$.

In the monoidal category $(\text{Pfn}, +, 0)$. 

Pipe diagram

String diagram
Traced Sym. Monoidal Category 
\((\text{Pfn}, +, 0)\)

* Category Pfn of partial functions

* Obj. A set \(X\)

* Arr. A partial function

\[ X \rightarrow Y \text{ in Pfn} \]

\[ X \leftarrow Y, \text{ partial function} \]
Traced Sym. Monoidal Category (Pfn, +, 0)

* Category Pfn of partial functions

* Obj. A set \( X \)

* Arr. A partial function

\[
\begin{array}{c}
X 
\rightarrow Y \text{ in Pfn} \\
X 
\rightarrow Y, \text{ partial function}
\end{array}
\]

* is traced symmetric monoidal
Traced Sym. Monoidal Category
\((Pfn, +, 0)\)

\[
\begin{align*}
X + Z \xrightarrow{f} Y + Z & \quad \text{in } Pfn \\
X \xrightarrow{\text{tr}(f)} Y & \quad \text{in } Pfn
\end{align*}
\]

How?
Traced Sym. Monoidal Category
\[(Pfn, +, 0)\]

\[
\begin{align*}
X + Z & \xrightarrow{f} Y + Z & \text{in } Pfn \\
X & \xrightarrow{\text{tr}(f)} Y & \text{in } Pfn
\end{align*}
\]
Traced Sym. Monoidal Category

\((\text{Pfn}, +, 0)\)

\[
\begin{align*}
X + Z & \xrightarrow{f} Y + Z \quad \text{in Pfn} \\
X & \xrightarrow{\text{tr}(f)} Y \quad \text{in Pfn}
\end{align*}
\]
Traced Sym. Monoidal Category
(Pfn, +, 0)

\[ X + Z \xrightarrow{f} Y + Z \quad \text{in Pfn} \]
\[ X \xrightarrow{\text{tr}(f)} Y \quad \text{in Pfn} \]

**How?**

\[ f_{XY}(x) := \begin{cases} f(x) & \text{if } f(x) \in Y \\ \bot & \text{o.w.} \end{cases} \]

Similar for \( f_{XZ}, f_{ZY}, f_{ZZ} \)
Traced Sym. Monoidal Category (Pfn, +, 0)

\[
X + Z \xrightarrow{f} Y + Z \quad \text{in Pfn}
\]

\[
X \xrightarrow{\text{tr}(f)} Y \quad \text{in Pfn}
\]

How?

\[
f_{XY}(x) := \begin{cases} 
    f(x) & \text{if } f(x) \in Y \\
    \bot & \text{o.w.}
\end{cases}
\]

Similar for \( f_{XZ}, f_{ZY}, f_{ZZ} \)

Trace operator:
Traced Sym. Monoidal Category

\[(\text{Pfn}, +, 0)\]

\[\begin{align*}
X + Z & \overset{f}{\rightarrow} Y + Z \quad \text{in Pfn} \\
X & \overset{\text{tr}(f)}{\rightarrow} Y \quad \text{in Pfn}
\end{align*}\]

How?

\[f_{XY}(x) := \begin{cases} 
  f(x) & \text{if } f(x) \in Y \\
  \bot & \text{o.w.}
\end{cases}\]

Similar for \(f_{XZ}, f_{ZY}, f_{ZZ}\)

Trace operator:

\[\text{tr}(f) = f_{XY} \sqcup \left( \coprod_{n \in \mathbb{N}} f_{ZY} \circ (f_{ZZ})^n \circ f_{XZ} \right)\]
**Traced Sym. Monoidal Category**

\[(Pfn, +, 0)\]

\[
\begin{align*}
X + Z & \xrightarrow{f} Y + Z \quad \text{in } Pfn \\
X & \xrightarrow{\text{tr}(f)} Y \quad \text{in } Pfn
\end{align*}
\]

**Trace operator:**

\[
\begin{align*}
\text{tr}(f) = \\
f_{XY} \sqcup \left( \coprod_{n \in \mathbb{N}} f_{ZY} \circ (f_{ZZ})^n \circ f_{XZ} \right)
\end{align*}
\]

\[f_{XY}(x) :=
\begin{cases}
  f(x) & \text{if } f(x) \in Y \\
  \perp & \text{o.w.}
\end{cases}
\]

Similar for \(f_{XZ}, f_{ZY}, f_{ZZ}\)

- **Execution formula** (Girard)
- Partiality is essential (infinite loop)
**GoI situation**

* Traced sym. monoidal cat.

* Where one can “feedback”

* Why for GoI?

---

**Defn. (GoI situation [AHS02])**

A GoI situation is a triple $(\mathcal{C}, F, U)$ where

- $\mathcal{C} = (\mathcal{C}, \otimes, I)$ is a traced symmetric monoidal category (TSMC);
- $F : \mathcal{C} \to \mathcal{C}$ is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).
  - $e : FF < F : e'$ (Comultiplication)
  - $d : \text{id} < F : d'$ (Dereliction)
  - $c : F \otimes F < F : c'$ (Contraction)
  - $w : K_I < F : w'$ (Weakening)

Here $K_I$ is the constant functor into the monoidal unit $I$;

- $U \in \mathcal{C}$ is an object (called reflexive object), equipped with the following retractions.
  - $j : U \otimes U < U : k$
  - $I < U$
  - $u : FU < U : v$
\[ MN \] = \begin{bmatrix} M \end{bmatrix} \begin{bmatrix} N \end{bmatrix} \]
\[
\begin{bmatrix}
M \\
N
\end{bmatrix}
= \begin{bmatrix}
M \\
\end{bmatrix}
\begin{bmatrix}
N
\end{bmatrix}
\text{ in string diagram}
\]
**GoI situation**

*Traced sym. monoidal cat.*

*Where one can “feedback”*

*Why for GoI?*

*Leading example: Pfn*
**Defn.** (GoI situation [AHS02])
A GoI situation is a triple \((C, F, U)\) where

- \(C = (C, \otimes, I)\) is a traced symmetric monoidal category (TSMC);
- \(F : C \rightarrow C\) is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).
  - \(e : FF \triangleleft F : e'\) (Comultiplication)
  - \(d : id \triangleleft F : d'\) (Dereliction)
  - \(c : F \otimes F \triangleleft F : c'\) (Contraction)
  - \(w : K_I \triangleleft F : w'\) (Weakening)

Here \(K_I\) is the constant functor into the monoidal unit \(I\).
- \(U \in C\) is an object (called reflexive object), equipped with the following retractions.
  - \(j : U \otimes U \triangleleft U : k\)
  - \(I \triangleleft U\)
  - \(u : FU \triangleleft U : v\)

**Defn.** (Retraction)
A retraction from \(X\) to \(Y\),
\[
f : X \triangleleft Y : g,
\]
is a pair of arrows
\[
\text{id} 
\begin{array}{c}
\circlearrowleft F \\
\circlearrowright \\
g
\end{array}
\text{id}
\]
such that \(g \circ f = \text{id}_X\).

\* **Functor** \(F\)

\* For obtaining \(!: A \rightarrow A\)
**GoI situation**

**Defn.** (GoI situation [AHS02])

A GoI situation is a triple \((\mathcal{C}, F, U)\) where

- \(\mathcal{C} = (\mathcal{C}, \otimes, I)\) is a traced symmetric monoidal category (TSMC);
- \(F : \mathcal{C} \to \mathcal{C}\) is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).
  
  - \(e : FF \trianglelefteq F : e'\) Comultiplication
  - \(d : \text{id} \trianglelefteq F : d'\) Dereliction
  - \(c : F \otimes F \trianglelefteq F : c'\) Contraction
  - \(w : K_I \trianglelefteq F : w'\) Weakening

Here \(K_I\) is the constant functor into the monoidal unit \(I\).

- \(U \in \mathcal{C}\) is an object (called reflexive object), equipped with the following retractions.

  - \(j : U \otimes U \trianglelefteq U : k\)
  - \(I \trianglelefteq U\)
  - \(u : FU \trianglelefteq U : v\)
**GoI situation**

**Defn.** (GoI situation [AHS02])

A GoI situation is a triple $(C, F, U)$ where

- $C = (C, \otimes, I)$ is a traced symmetric monoidal category (TSMC);
- $F : C \to C$ is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).
  
  $e : FF \otimes F \to F' \quad$ Comultiplication
  
  $d : \text{id} \otimes F \to F' \quad$ Dereliction
  
  $c : F \otimes F \otimes F \to F' \quad$ Contraction
  
  $w : K_I \otimes F \to F' \quad$ Weakening

Here $K_I$ is the constant functor into the monoidal unit $I$.

- $U \in C$ is an object (called reflexive object), equipped with the following retractions.

  $j : U \otimes U \to U \otimes U \quad \text{with} \quad k = \text{id}$

- $\text{Retr. } U \otimes U \to U \quad j \quad k$
GoI situation

Defn. (GoI situation [AHS02])
A GoI situation is a triple \((\mathbb{C}, F, U)\) where

- \(\mathbb{C} = (\mathbb{C}, \otimes, I)\) is a traced symmetric monoidal category (TSMC);
- \(F : \mathbb{C} \to \mathbb{C}\) is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).
  
  - \(e : FF \triangleleft F : e'\) (Comultiplication)
  - \(d : \text{id} \triangleleft F : d'\)
  - \(c : F \otimes F \triangleleft F : c'\)
  - \(w : K_I \triangleleft F : w'\)

  Here \(K_I\) is the constant functor.

- \(U \in \mathbb{C}\) is an object (called reflexive object), equipped with the following retractions.
  
  - \(j : U \otimes U \triangleleft U : k\)
  - \(I \triangleleft U\)
  - \(u : FU \triangleleft U : v\)

\* The reflexive object \(U\)

\* Why for GoI?

\* Example in Pfn:
**GoI situation**

* The reflexive object $U$

* Why for GoI?

* Example in Pfn:

---

**Defn.** (GoI situation [AHS02])
A GoI situation is a triple $(C, F, U)$ where

- $C = (C, \otimes, I)$ is a traced symmetric monoidal category (TSMC);
- $F : C \to C$ is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).
  - $e : FF \triangleleft F : e'$ Comultiplication
  - $d : \text{id} \triangleleft F : d'$
  - $c : F \otimes F \triangleleft F : c'$ Dereliction
  - $w : K_I \triangleleft F : w'$ Contraction
  - $K_I$ is the constant functor
- $U \in C$ is an object (called reflexive object), equipped with the following retractions.
  - $j : U \otimes U \triangleleft U : k$
  - $I \triangleleft U$
  - $u : FU \triangleleft U : v$

---

Hasuo (Tokyo)
GoI situation

**Defn.** (GoI situation [AHS02])

A GoI situation is a triple $(C, F, U)$ where

- $C = (C, \otimes, I)$ is a traced symmetric monoidal category (TSMC);
- $F : C \to C$ is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).
  
  - $e : FF \bowtie F : e'$
  - $d : id \bowtie F : d'$
  - $c : F \otimes F \bowtie F : c'$
  - $w : K_I \bowtie F : w'$

Here $K_I$ is the constant functor.

- $U \in C$ is an object (called *reflexive object*), equipped with the following retractions.
  
  - $j : U \otimes U \bowtie U : k$
  - $I \bowtie U$
  - $u : FU \bowtie U : v$

**Why for GoI?**

**Example in Pfn:**

$N \in \text{Pfn}$, with

- $N + N \cong N$,
- $N \cdot N \cong N$
**GoI Situation: Summary**

* Categorical axiomatics of the “GoI animation”

---

**Defn. (GoI situation [AHS02])**

A GoI situation is a triple \((C, F, U)\) where

- \(C = (C, \otimes, I)\) is a traced symmetric monoidal category (TSMC);
- \(F : C \to C\) is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).
  - \(e : FF \triangleleft F \triangleleft e'\)  
    Comultiplication
  - \(d : \text{id} \triangleleft F \triangleleft d'\)  
    Dereliction
  - \(c : F \otimes F \triangleleft F \triangleleft c'\)  
    Contraction
  - \(w : K_I \triangleleft F \triangleleft w'\)  
    Weakening

Here \(K_I\) is the constant functor into the monoidal unit \(I\);

- \(U \in C\) is an object (called reflexive object), equipped with the following retractions.
  - \(j : U \otimes U \triangleleft U \triangleleft k\)  
    \(I \triangleleft U\)
  - \(u : FU \triangleleft U \triangleleft v\)

---

**Example:**

\((\text{Pfn}, N \cdot _\_, N)\)
Defn. (GoI situation [AHS02])
A GoI situation is a triple $(A, F, U)$ where

- $C = (C, \otimes, I)$ is a traced symmetric monoidal category (TSMC);
- $F : C \to C$ is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).
  - $e : FF < F : e'$ Comultiplication
  - $d : id < F : d'$ Dereliction
  - $c : F \otimes F < F : c'$ Contraction
  - $w : K_I < F : w'$ Weakening

Here $K_I$ is the constant functor into the monoidal unit $I$;
- $U \in C$ is an object (called reflexive object), equipped with the following retractions.
  - $j : U \otimes U < U : k$
  - $I < U$
  - $u : FU < U : v$

Example:

$(\text{Pfn}, \ N \cdot \_ \ , \ N)$
**Categorical axiomatics of the "GoI animation"**

**Example:**

\[(\text{Pfn}, N \cdot \_\_ N)\]
**Categorical axiomatics of the “GoI animation”**

**Definition (GoI situation [AHS02])**

A GoI situation is a triple \((C, F, U)\) where:

- \(C = (C, \otimes, I)\) is a traced symmetric monoidal category (TSMC);
- \(F : C \to C\) is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations):
  - \(e : FF \otimes F \to F : e'\) (Dereliction)
  - \(d : \text{id} \otimes F \to F : d'\) (Comultiplication)
  - \(c : F \otimes F \otimes F \to F : c'\) (Contraction)
  - \(w : K_I \otimes F \to F : w'\) (Weakening)

Here \(K_I\) is the constant functor into the monoidal unit \(I\);

- \(U \in C\) is an object (called reflexive object), equipped with the following retractions:
  - \(j : U \otimes U \to U : k\)
  - \(u : FU \to U : v\)

For \(!\), via

\[f \xrightarrow{F} f\]

**Example:**

\[(\text{Pfn}, N \cdot _{-}, N)\]
Categorical axiomatics of the "GoI animation"

Example:

(Pfn, \( N \cdot _{-} \cdot N \))
Categorical GoI: Constr. of an LCA

**Thm.** ([AHS02])
Given a GoI situation \((\mathcal{C}, F, U)\), the homset

\[ \mathcal{C}(U, U) \]

carries a canonical LCA structure.
**Thm. ([AHS02])**
Given a GoI situation \((\mathcal{C}, F, U)\), the homset \(\mathcal{C}(U, U)\) carries a canonical LCA structure.

- Applicative str. •
- ! operator
- Combinators B, C, I, ...
Categorical GoI: Constr. of an LCA

**Thm.** ([AHS02])
Given a GoI situation \((\mathbb{C}, F, U)\), the homset \(\mathbb{C}(U, U)\) carries a canonical LCA structure.

- Applicative str. \(\cdot\)
- ! operator
- Combinators B, C, I, ...

\[
\begin{array}{c}
U \\
\downarrow \\
U \\
\end{array}
\xrightarrow{f} \in \mathbb{C}(U, U)
\]
**Categorical GoI: Constr. of an LCA**

**Thm. ([AHS02])**

Given a GoI situation \((\mathcal{C}, F, U)\), the homset \(\mathcal{C}(U, U)\) carries a canonical LCA structure.

- **Applicative str. \cdot**
- **! operator**
- **Combinators B, C, I, ...**

\[
g \cdot f := \text{tr}( (U \otimes f) \circ k \circ g \circ j )
\]
**Thm.** ([AHS02])
Given a GoI situation \((C, F, U)\), the homset \(C(U, U)\) carries a canonical LCA structure.

\[ \text{Applicative str. } \cdot \]
\[ \text{! operator} \]
\[ \text{Combinators } B, C, I, \ldots \]

\[ ! f := u \circ Ff \circ v \]

Pipe diagram
Categorical GoI: Constr. of an LCA

* Combinator \( B_{xyz} = x(yz) \)

Figure 7: Composition Combinator B

from [AHS02]
Categorical GoI: Constr. of an LCA

* Combinator \( B_{xyz} = x(yz) \)
Hasuo (Tokyo)

Categorical GoI: Constr. of an LCA

Combinator

\[ B_{xyz} = x(yz) \]
Categorical GoI: Constr. of an LCA

\[ Bxyz = x(yz) \]
Categorical GoI: Constr. of an LCA

\[ Bxyz = x(yz) \]
Categorical GoI: Constr. of an LCA

* Combinator \( Bxyz = x(yz) \)

Figure 7: Composition Combinator B

from [AHS02]
Categorical GoI: Constr. of an LCA

* Combinator \( B_{xyz} = x(yz) \)

Nice dynamic interpretation of (linear) computation!!

Figure 7: Composition Combinator B from [AHS02]
Summary: Categorical GoI

**Defn.** (GoI situation [AHS02])

A *GoI situation* is a triple $(\mathcal{C}, F, U)$ where

- $\mathcal{C} = (\mathcal{C}, \otimes, I)$ is a traced symmetric monoidal category (TSMC);
- $F : \mathcal{C} \to \mathcal{C}$ is a traced symmetric monoidal functor, equipped with the following retractions (which are monoidal natural transformations).
  - $e : FF \triangleright F : e'$ \hspace{1cm} Comultiplication
  - $d : \text{id} \triangleleft F : d'$ \hspace{1cm} Dereliction
  - $c : F \otimes F \triangleright F : c'$ \hspace{1cm} Contraction
  - $w : K_I \triangleright F : w'$ \hspace{1cm} Weakening

Here $K_I$ is the constant functor into the monoidal unit $I$;
- $U \in \mathcal{C}$ is an object (called *reflexive object*), equipped with the following retractions.
  - $j : U \otimes U \triangleleft U : k$
  - $I \triangleleft U$
  - $u : FU \triangleright U : v$

**Thm.** ([AHS02])

Given a GoI situation $(\mathcal{C}, F, U)$, the homset $\mathcal{C}(U, U)$ carries a canonical LCA structure.
Why Categorical Generalization?: Examples Other Than Pfn [AHS02]

* Strategy: find a TSMC!

* "Wave-style" examples
  * $\otimes$ is Cartesian product(-like)
  * in which case,

$$\text{trace} \approx \text{fixed point operator} \quad \text{[Hasegawa/Hyland]}$$

* An example:
  $$((\omega\text{-Cpo}, \times, 1), (\_)^N, A^N)$$

* (... less of a dynamic flavor)
Why Categorical Generalization?: Examples Other Than Pfn [AHS02]

* "Particle-style" examples
* Obj. $X \in \mathcal{C}$ is set-like; $\otimes$ is coproduct-like
* The GoI animation is valid

* Examples:
  * Partial functions
    $((\text{Pfn}, +, 0), \mathbb{N} \cdot \_ , \mathbb{N})$
  * Binary relations
    $((\text{Rel}, +, 0), \mathbb{N} \cdot \_ , \mathbb{N})$
  * "Discrete stochastic relations"
    $((\text{DSRel}, +, 0), \mathbb{N} \cdot \_ , \mathbb{N})$
Why Categorical Generalization?:
Examples Other Than Pfn \cite{AHS02}

- **Pfn (partial functions)**
  
  \[
  \begin{align*}
  &X \to Y \text{ in } \text{Pfn} \\
  \Rightarrow &X \to Y, \text{ partial function} \\
  \Rightarrow &X \to \mathcal{L}Y \text{ in } \text{Sets}
  \end{align*}
  \]
  where \( \mathcal{L}Y = \{ \bot \} + Y \)

- **Rel (relations)**
  
  \[
  \begin{align*}
  &X \to Y \text{ in } \text{Rel} \\
  \Rightarrow &R \subseteq X \times Y, \text{ relation} \\
  \Rightarrow &X \to \mathcal{P}Y \text{ in } \text{Sets}
  \end{align*}
  \]
  where \( \mathcal{P} \) is the powerset monad

- **DSRel**
  
  \[
  \begin{align*}
  &X \to Y \text{ in } \text{DSRel} \\
  \Rightarrow &X \to \mathcal{D}Y \text{ in } \text{Sets} \\
  &\text{where } \mathcal{D}Y = \{ d : Y \to [0, 1] \mid \sum_y d(y) \leq 1 \}
  \end{align*}
  \]
Why Categorical Generalization?

Examples Other Than Pfn [AHS02]

* **Pfn** (partial functions)

\[
\begin{align*}
X \to Y & \text{ in Pfn} \\
X \rightarrow Y, \text{ partial function} & \\
X \to \mathcal{L}Y & \text{ in Sets} \\
\end{align*}
\]

where \( \mathcal{L}Y = \{ \bot \} + Y \)

* **Rel** (relations)

\[
\begin{align*}
X \to Y & \text{ in Rel} \\
R \subseteq X \times Y, \text{ relation} & \\
X \to \mathcal{P}Y & \text{ in Sets} \\
\end{align*}
\]

where \( \mathcal{P} \) is the powerset monad

* **DSRel**

\[
\begin{align*}
X \to Y & \text{ in DSRel} \\
X \to \mathcal{D}Y & \text{ in Sets} \\
\end{align*}
\]

where \( \mathcal{D}Y = \{ d : Y \to [0, 1] \mid \sum_y d(y) \leq 1 \} \)

Categories of sets and (functions with different branching/partiality)

Hasuo (Tokyo)

Tuesday, October 9, 12
Why Categorial Generalization? Examples Other Than Pfn [AHS02]

* **Pfn** (partial functions)
  \[
  \frac{X \rightarrow Y \text{ in } \text{Pfn}}{X \rightarrow Y, \text{ partial function}} \quad \text{where } \mathcal{L}Y = \{\bot\} + Y
  \]

* **Rel** (relations)
  \[
  \frac{X \rightarrow Y \text{ in } \text{Rel}}{R \subseteq X \times Y, \text{ relation}} \quad \text{where } \mathcal{P} \text{ is the powerset monad}
  \]

* **DSRel**
  \[
  \frac{X \rightarrow Y \text{ in DSRel}}{X \rightarrow \mathcal{D}Y \text{ in Sets}} \quad \text{where } \mathcal{D}Y = \{d : Y \rightarrow [0, 1] \mid \sum_y d(y) \leq 1\}
  \]

(Potential) non-termination

Non-determinism

Probabilistic branching
Different Branching in The GoI Animation

- **Pfn** (partial functions)
- Pipes can be stuck
- **Rel** (relations)
- Pipes can branch
- **DSRel**
- Pipes can branch probabilistically

---

Hasuo (Tokyo)

Tuesday, October 9, 12
Different Branching in The GoI Animation

- Pfn (partial functions)
  - Pipes can be stuck

- Rel (relations)
  - Pipes can branch

- DSRel
  - Pipes can branch probabilistically
Different Branching in The GoI Animation

- **Pfn** (partial functions)
- Pipes can be stuck
- **Rel** (relations)
- Pipes can branch
- **DSRel**
- Pipes can branch probabilistically
Different Branching in The GoI Animation

- \texttt{Pfn} (partial functions)
- Pipes can be stuck
- \texttt{Rel} (relations)
- Pipes can branch
- \texttt{DSRel}
- Pipes can branch probabilistically
Different Branching in The GoI Animation

- **Pfn** (partial functions)
- Pipes can be stuck
- **Rel** (relations)
- Pipes can branch
- **DSRel**
- Pipes can branch probabilistically
Different Branching in The GoI Animation

* **Pfn** (partial functions)
* Pipes can be stuck
* **Rel** (relations)
* Pipes can branch
* **DSRel**
  * Pipes can branch probabilistically
Different Branching in The GoI Animation

- \textbf{Pfn} (partial functions)
- Pipes can be stuck
- \textbf{Rel} (relations)
- Pipes can branch
- \textbf{DSRel}
- Pipes can branch probabilistically
Why Categorical Generalization?
Examples Other Than Pfn

* **Pfn** (partial functions)

\[
\begin{align*}
X & \to Y \text{ in } \text{Pfn} \\
X & \to Y, \text{ partial function} \\
X & \to \mathcal{L}Y \text{ in } \text{Sets}
\end{align*}
\]

\[
\text{where } \mathcal{L}Y = \{ \bot \} + Y
\]

* **Rel** (relations)

\[
\begin{align*}
X & \to Y \text{ in } \text{Rel} \\
R & \subseteq X \times Y, \text{ relation} \\
X & \to \mathcal{P}Y \text{ in } \text{Sets}
\end{align*}
\]

\[
\text{where } \mathcal{P} \text{ is the powerset monad}
\]

* **DSRel**

\[
\begin{align*}
X & \to Y \text{ in } \text{DSRel} \\
X & \to \mathcal{D}Y \text{ in } \text{Sets}
\end{align*}
\]

\[
\text{where } \mathcal{D}Y = \{ d : Y \to [0, 1] | \sum_y d(y) \leq 1 \}
\]

Categories of sets and (functions with different branching/partiality)

(Potential) non-termination

Non-determinism

Probabilistic branching
Why Categorical Generalization?

Examples Other Than Pfn

- **Pfn (partial functions)**
  \[
  X \rightarrow Y \in \text{Pfn} \\
  X \rightarrow Y, \text{ partial function} \\
  X \rightarrow \mathcal{L}Y \in \text{Sets}
  \]
  where \( \mathcal{L}Y = \{ \bot \} + Y \)

- **Rel (relations)**
  \[
  X \rightarrow Y \in \text{Rel} \\
  R \subseteq X \times Y, \text{ relation} \\
  X \rightarrow \mathcal{P}Y \in \text{Sets}
  \]
  where \( \mathcal{P} \) is the powerset monad

- **DSRel**
  \[
  X \rightarrow Y \in \text{DSRel} \\
  X \rightarrow \mathcal{D}Y \in \text{Sets}
  \]
  where \( \mathcal{D}Y = \{ d : Y \rightarrow [0, 1] \mid \sum_y d(y) \leq 1 \} \)

\( KL(B) \) for different branching monads \( B \)

(Potential) non-termination
Non-determinism
Probabilistic branching

Categories of sets and \((\text{functions with different branching/partiality})\)

(Kl(B)) for different branching monads \( B \)

Hasuo (Tokyo)
Part 2

Coalgebraic Trace Semantics
Trace Semantics of Systems

\[ \text{tr}(x) = \{a, ab, abb, \ldots\} = ab^* \]

* Non-deterministic branching: sign. functor is \( P(1 + \Sigma \times \_\)\)
Bisimilarity vs. Trace Sem.
Bisimilarity vs. Trace Sem.
Bisimilarity vs. Trace Sem.
**Bisimilarity**

Branching structure matters.
Can I choose later?

**Trace semantics**

Branching structure does not matter.
Anyway we’ll get the same sets of food.
**Bisimilarity vs. Trace Sem.**

**Bisimilarity**
Branching structure matters.
Can I choose later?

**Trace semantics**
Branching structure does not matter.
Anyway we’ll get the same sets of food.

Also by final coalgebra?

\[
\begin{align*}
F \overset{\text{beh}(c)}{\longrightarrow} & \\
FX \overset{\text{beh}(c)}{\longrightarrow} & \overset{\text{beh}(c)}{\longrightarrow} FZ \\
X \overset{\text{beh}(c)}{\longrightarrow} & \overset{\text{beh}(c)}{\longrightarrow} Y \\
\end{align*}
\]
**Coinduction in a Kleisli Category**

[ IH, Jacobs, Sokolova, '07 ]

\[
\begin{align*}
X & \rightarrow Y \quad \text{in } \mathcal{K}(B) \\
& \rightarrow BY \quad \text{in } \text{Sets}
\end{align*}
\]

**Thm.** Let \( F \) be an endofunctor, and \( B \) be a monad, both on \( \text{Sets} \). Assume:

1. We have a distributive law \( \lambda : FB \Rightarrow BF \).
2. The functor \( F \) preserves \( \omega \)-colimits, yielding an initial algebra \( \frac{FA}{\alpha} \).
3. The Kleisli category \( \mathcal{K}(B) \) is \( \text{Cpo}_{\perp} \)-enriched and composition in \( \mathcal{K}(B) \) is left-strict.

Then:

1. \( F \) lifts to \( \overline{F} : \mathcal{K}(B) \rightarrow \mathcal{K}(B) \), with \( JF = \overline{FJ} \).
2. \( \frac{\eta \circ \alpha}{A} \) is an initial algebra in \( \mathcal{K}(B) \).
3. In \( \mathcal{K}(B) \) we have initial algebra-final coalgebra coincidence and \( \frac{FA}{\alpha} \) is a final coalgebra.
Coinduction in a Kleisli Category

[IH, Jacobs, Sokolova, ’07]

\[ X \rightarrow Y \quad \text{in} \quad \mathcal{K}l(B) \]

\[ X \rightarrow BY \quad \text{in} \quad \text{Sets} \]

* Initial algebra lifts from \( \text{Sets} \) to \( Kl(B) \)

* diagram chasing [Johnstone]

**Thm.** Let \( F \) be an endofunctor, and \( B \) be a monad, both on \( \text{Sets} \). Assume:

1. We have a distributive law \( \lambda : FB \Rightarrow BF \).
2. The functor \( F \) preserves \( \omega \)-colimits, yielding an initial algebra \( \frac{FA}{A} \approx \downarrow \alpha \).
3. The Kleisli category \( \mathcal{K}l(B) \) is \( \text{Cpo}_\perp \)-enriched and composition in \( \mathcal{K}l(B) \) is left-strict.

Then:

1. \( F \) lifts to \( \overline{F} : \mathcal{K}l(B) \rightarrow \mathcal{K}l(B) \), with \( JF = \overline{F}J \).
2. \( \frac{AF}{A} \eta \circ \alpha \) is an initial algebra in \( \mathcal{K}l(B) \).
3. In \( \mathcal{K}l(B) \) we have initial algebra-final coalgebra coincidence and \( \frac{FA}{A} (\eta \circ \alpha)^{-1} \) is a final coalgebra.
Coinduction in a Kleisli Category

* Initial algebra lifts from Sets to $\mathcal{Kl}(B)$
  * diagram chasing [Johnstone]

* In $\mathcal{Kl}(B)$ we have IA–FC coincidence
  * typical of “domain-theoretic” categories
  * “Algebraically compact” [Freyd]

**Thm.** Let $F$ be an endofunctor, and $B$ be a monad, both on Sets. Assume:

1. We have a distributive law $\lambda : FB \Rightarrow BF$.
2. The functor $F$ preserves $\omega$-colimits, yielding an initial algebra $\alpha : FA \overset{\approx}{\to} A$.
3. The Kleisli category $\mathcal{Kl}(B)$ is $\mathbf{Cpo}_\bot$-enriched and composition in $\mathcal{Kl}(B)$ is left-strict.

Then:

1. $F$ lifts to $\overline{F} : \mathcal{Kl}(B) \to \mathcal{Kl}(B)$, with $JF = \overline{F}J$.
2. $\overline{\eta} \circ \alpha$ is an initial algebra in $\mathcal{Kl}(B)$.
3. In $\mathcal{Kl}(B)$ we have initial algebra-final coalgebra coincidence and $\overline{F} \overline{(\eta \circ \alpha)}^{-1}$ is a final coalgebra.
Coinduction in a Kleisli Category

* E.g. \( B = \mathcal{P}, \ F = 1 + \Sigma \times (\_) \)

\[
\begin{align*}
1 + \Sigma \times X & \rightarrow 1 + \Sigma \times \Sigma^* \\
\begin{array}{c}
\xymatrix{c & X \ar[l] & 1 + \Sigma \times \text{tr}(c) \\
\text{tr}(c) & \Sigma^* \ar[l] & \text{final} \ar[l]}
\end{array}
\end{align*}
\]

* Separation between \( B \) and \( F \)

Hasuo (Tokyo)
Coinduction in a Kleisli Category

* E.g. \( B = \mathcal{P} \), \( F = 1 + \Sigma \times (\_\_) \)

\[
\begin{array}{c}
P(1 + \Sigma \times X) \\
c \uparrow \\
X
\end{array} \rightarrow \begin{array}{c}
1 + \Sigma \times X \\
1 + \Sigma \times \text{tr}(c) \\
\text{tr}(c)
\end{array} \rightarrow \begin{array}{c}
1 + \Sigma \times \Sigma^* \\
\text{final} \in \mathcal{Kl}(\mathcal{P})
\end{array}

* Separation between \( B \) and \( F \)
Coinduction in a Kleisli Category

* E.g. $B = \mathcal{P}$, $F = 1 + \Sigma \times (\_)$

\[
\begin{align*}
1 + \Sigma \times X & \xrightarrow{\cdot c} 1 + \Sigma \times \text{tr}(c) \\
\downarrow & \\
\Sigma^* & \xrightarrow{\text{final}} \mathcal{K}(\mathcal{P})
\end{align*}
\]

* Separation between $B$ and $F$
Coinduction in a Kleisli Category

* E.g. $B = \mathcal{P}$, $F = 1 + \Sigma \times (\_)$

\[
1 + \Sigma \times X \xrightarrow{\Sigma} 1 + \Sigma \times \Sigma^* \xrightarrow{\text{final}} \Sigma^* \xrightarrow{\text{initial}} \text{in } \mathcal{K} \ell(\mathcal{P})
\]

\[
\mathcal{P}(1 + \Sigma \times X) \xrightarrow{c} \text{in Sets}
\]

\[
X \xrightarrow{\text{tr}(c)} \Sigma^* \xrightarrow{\mathcal{P}} \mathcal{P}(\Sigma^*)
\]

* Separation between $B$ and $F$

Hasuo (Tokyo)
Coinduction in a Kleisli Category

* E.g. $B = \mathcal{P}$, $F = 1 + \Sigma \times (\_)$

\[
\begin{array}{c}
1 + \Sigma \times X \\
\uparrow c \\
X
\end{array} \quad \xrightarrow{\text{---} + + - -} \quad \begin{array}{c}
1 + \Sigma \times \Sigma^* \\
\uparrow \text{final} \\
\Sigma^*
\end{array}
\]

\[
\begin{array}{c}
\mathcal{P}(1 + \Sigma \times X) \\
\uparrow c \\
X
\end{array} \quad \xrightarrow{\text{---} + + - -} \quad \begin{array}{c}
\Sigma^* \\
\uparrow \text{final} \\
\mathcal{P}(\Sigma^*)
\end{array}
\]

* Separation between $B$ and $F$

induced by $1 + \Sigma \times \Sigma^*$, initial $\Sigma^*$, in $\text{Sets}$

$\text{tr}(c)$

\[
\text{tr}(x) = \{a, ab, abb, \ldots\} = ab^*
\]
Coinduction in a Kleisli Category

* E.g. \( B = \mathcal{P}, \ F = 1 + \Sigma \times (_) \)

\[
\begin{align*}
\mathcal{P}(1 + \Sigma \times X) & \xrightarrow{c} X \\
\downarrow \quad \downarrow \quad \downarrow \quad \downarrow \\
1 + \Sigma \times \text{tr}(c) & \rightarrow 1 + \Sigma \times \Sigma^* \\
\uparrow c & \quad \uparrow \text{final} & \uparrow \text{initial} \\
\Sigma^* & \rightarrow \mathcal{K}\ell(\mathcal{P}) \\
\text{tr}(c) & \rightarrow \mathcal{P}(\Sigma^*) \\
\end{align*}
\]

* Separation between \( B \) and \( F \)

\[
\text{tr}(x) = \{a, ab, abb, \ldots \} = ab^*
\]

Hasuo (Tokyo)
Examples

* A branching monad $B$:
  * Lift monad $\mathcal{L} = 1 + (\_)$, powerset monad $\mathcal{P}$, subdistribution monad $\mathcal{D}$
  * Precisely those in $\text{A functor } F$: polynomial functors
The Coauthor

* Naohiko Hoshino
* DSc (Kyoto, 2011)
* Supervisor: Masahito “Hassei” Hasegawa
* Currently at RIMS, Kyoto U.
* http://www.kurims.kyoto-u.ac.jp/~naophiko/
Thm. ([Jacobs,CMCS10])
Given a “branching monad” $B$ on $\text{Sets}$, the
monoidal category

$$(\mathcal{K}\ell(B), +, 0)$$

is a traced symmetric monoidal category.

Cor.
$$( (\mathcal{K}\ell(B), +, 0), \mathbb{N} \cdot \_ \cdot \mathbb{N} )$$
is a GoI situation.
Thm. ([Jacobs,CMCS10])
Given a "branching monad" $B$ on $\text{Sets}$, the monoidal category

$$(\mathcal{K}\ell(B), +, 0)$$

is a traced symmetric monoidal category.

Cor.
$$( (\mathcal{K}\ell(B), +, 0), \mathbb{N} \cdot \_, \mathbb{N} )$$

is a GoI situation.
Thm. ([Jacobs,CMCS10])
Given a “branching monad” $B$ on $\text{Sets}$, the monoidal category

$$(\mathcal{Kl}(B), +, 0)$$

is a traced symmetric monoidal category.

Cor.

$$( (\mathcal{Kl}(B), +, 0), N \cdot _, N )$$ is a GoI situation.

Proof. We need

$$\begin{align*}
X + Z \xrightarrow{f} Y + Z & \quad \text{in } \mathcal{Kl}(T) \\
X \xrightarrow{\text{tr}(f)} Y & \quad \text{in } \mathcal{Kl}(T)
\end{align*}$$

- $X + Z \xrightarrow{f} Y + Z \xrightarrow{\kappa} Y + (X + Z)$ is a $Y + (\_ )$-coalgebra

- $Y + N \cdot Y$ is an initial algebra in $\text{Sets}$ $N \cdot Y$

- Therefore in $\mathcal{Kl}(T)$:

$$\begin{align*}
Y + (X + Z) \xrightarrow{\kappa \circ f} Y + N \cdot Y \\
X + Z \xrightarrow{\text{tr}(c)} N \cdot Y \\
\kappa_1 \xrightarrow{\text{final}} Y
\end{align*}$$

Hasuo (Tokyo)
From Coalgebraic Trace to Monoidal Trace

**Thm.** ([Jacobs,CMCS10])

Given a “branching monad” \( B \) on \( \text{Sets} \), the monoidal category

\[(\mathcal{K}_l(B), + , 0)\]

is a traced symmetric monoidal category.

**Cor.**

\(( (\mathcal{K}_l(B), + , 0), \mathbb{N} \cdot _- , \mathbb{N} ) \) is a GoI situation.

**Proof.** We need

\[
\begin{align*}
X + Z \xrightarrow{f} Y + Z \quad \text{in } \mathcal{K}_l(T) \\
X \xrightarrow{\text{tr}(f)} Y \quad \text{in } \mathcal{K}_l(T)
\end{align*}
\]

- \( X + Z \xrightarrow{f} Y + Z \xrightarrow{\kappa} Y + (X + Z) \)
  is a \( Y + (_-) \)-coalgebra

- \( Y + \mathbb{N} \cdot Y \)
  is an initial algebra in \( \text{Sets} \)

- \( \mathbb{N} \cdot Y \)
  is final

Therefore in \( \mathcal{K}_l(T) \):

\[
\begin{align*}
Y + (X + Z) \xrightarrow{f} Y + \mathbb{N} \cdot Y \\
\kappa \circ f \uparrow \quad \uparrow \text{final} \\
X + Z \xrightarrow{\kappa_1} \mathbb{N} \cdot Y \\
\kappa_1 \quad \uparrow \text{tr}(c) \quad \downarrow \nabla \\
X \quad Y
\end{align*}
\]
The Categorical GoI Workflow

Traced monoidal category $C$
+ other constructs $\leadsto$ "GoI situation" [AHS02]

Categorical GoI [AHS02]

Linear combinatory algebra

Realizability

Linear category
The Categorical GoI Workflow

- Branching monad $B$
- Coalgebraic trace semantics
- Traced monoidal category $\mathcal{C}$
  + other constructs $\Rightarrow$ "GoI situation" [AHS02]
- Categorical GoI [AHS02]
- Linear combinatory algebra
- Realizability
- Linear category
The Categorical GoI Workflow

Branching monad B

Coalgebraic trace semantics

Traced monoidal category $\mathbb{C}$
+ other constructs $\mapsto$ "GoI situation" [AHS02]

Categorical GoI [AHS02]

Linear combinatory algebra

Realizability

Linear category

Model of fancy language
The Categorical GoI Workflow

- Branching monad $B$
  - Coalgebraic trace semantics

- Traced monoidal category $\mathbb{C}$
  + Other constructs $\Rightarrow$ "GoI situation" [AHS02]

- Categorical GoI [AHS02]

- Linear combinatory algebra

- Realizability

- Linear category

Fancy LCA
Model of fancy language

Hasuo (Tokyo)
The Categorical GoI Workflow

- Linear category
- Realizability
- Linear combinatory algebra
- Categorical GoI [AHS02]
- Traced monoidal category \( \mathcal{C} \)
  + other constructs \( \rightarrow \) “GoI situation” [AHS02]
- Branching monad B
- Coalgebraic trace semantics

Fancy
- TSMC
- LCA
- Model of fancy language
The Categorical GoI Workflow

Branching monad $B$ ➜ Coalgebraic trace semantics

Traced monoidal category $\mathcal{C}$
+ other constructs ➔ “GoI situation” [AHS02]

Categorical GoI [AHS02]

Linear combinatory algebra

Realizability

Linear category

Fancy monad

Fancy TSMC

Fancy LCA

Model of fancy language

Tuesday, October 9, 12
What is Fancy, Nowadays?

Hasuo (Tokyo)
What is Fancy, Nowadays?

*Biology?
What is Fancy, Nowadays?

* Biology?
* Hybrid systems?
  * Both discrete and continuous data, typically in **cyber-physical systems (CPS)**
  * ➾ Our approach via **non-standard analysis**
    [Suenaga, IH, ICALP’11] [IH, Suenaga, CAV’12]
    [Suenaga, Sekine, IH, POPL’13]
What is Fancy, Nowadays?

* Biology?
* Hybrid systems?
  * Both discrete and continuous data, typically in *cyber-physical systems* (CPS)
  * Our approach via *non-standard analysis*
    [Suenaga, IH, ICALP’11] [IH, Suenaga, CAV’12]
    [Suenaga, Sekine, IH, POPL’13]
* Quantum?
* Yes this worked!
Future Directions

- GoI 2: Non-converging algs (untyped I-calc / PCF)
  - uses more topological info on operatn algs

- GoI 3: uses additives & additive proof nets —

- GoI 4 (last month): von Neumann algebras: $\mathcal{E}_X(f, z) \equiv f \
  \text{arb \ (not \ coming \ from \ proof)}$

- Quantum GoI?
Part 3

Future Directions

- GoI 2: Non-converging algms (untyped J-calc / PCF)
  - Uses more topological info on operatr algms

- GoI 3: uses additives & additive proof nets

- GoI 4 (last month): von Neumann algebras: $\mathbb{E}(f, x)$ $f_1 f$
  - $a b$ (not necessarily coming from proof)

- Quantum GoI?
The Categorical GoI Workflow

Branching monad $\mathbf{B}$

Coalgebraic trace semantics

Traced monoidal category $\mathcal{C}$
+ other constructs $\rightarrow$ “GoI situation” [AHS02]

Categorical GoI [AHS02]

Linear combinatory algebra

Realizability

Linear category
The Categorical GoI Workflow

- Branching monad $B$
- Coalgebraic trace semantics
- Traced monoidal category $\mathbb{C}$
  + other constructs $\Rightarrow$ "GoI situation" [AHS02]
- Categorical GoI [AHS02]
- Linear combinatory algebra
- Realizability
- Linear category

Quantum branching monad
Quantum TSMC
Quantum LCA
Model of quantum language
The Quantum Branching Monad

\[ QY = \left\{ c : Y \to \prod_{m,n \in \mathbb{N}} QO_{m,n} \mid \text{the trace condition} \right\} \]
The Quantum Branching

\[ QY = \left\{ c : Y \rightarrow \prod_{m,n \in \mathbb{N}} QO_{m,n} \mid \text{the trace condition} \right\} \]

\[ QO_{m,n} := \left\{ \text{quantum operations, from dim. } m \text{ to dim. } n \right\} \]
The Quantum Branching

\[
\mathbb{Q}_Y = \left\{ c : Y \to \prod_{m,n \in \mathbb{N}} \mathbb{Q} \right\}
\]

quantum operations, from dim. \( m \) to dim. \( n \)

the trace condition

\[
\sum_{y \in Y} \sum_{n \in \mathbb{N}} \text{tr}[(c(y))_{m,n}(\rho)] \leq 1 ,
\]

\[\forall m \in \mathbb{N}, \forall \rho \in D_m.\]
The Quantum Branching

\[ QY = \left\{ c : Y \to \prod_{m,n \in \mathbb{N}} QO_{m,n} \right\} \]

\[ QO_{m,n} := \left\{ \text{quantum operations, from dim. } m \text{ to dim. } n \right\} \]

the trace condition

\[ \sum_{y \in Y} \sum_{n \in \mathbb{N}} \text{tr}[(c(y))_{m,n}(\rho)] \leq 1, \forall m \in \mathbb{N}, \forall \rho \in D_m. \]

* Compare with

\[ \mathcal{P}Y = \left\{ c : Y \to 2 \right\} \]

\[ \mathcal{D}Y = \left\{ c : Y \to [0,1] \mid \sum_{y \in Y} c(y) \leq 1 \right\} \]

Hasuo (Tokyo)
The Quantum Branching Monad

\[ QY = \{ c : Y \rightarrow \prod_{m,n \in \mathbb{N}} QO_{m,n} \mid \text{the trace condition} \} \]

\[ QO_{m,n} := \{ \text{quantum operations, from dim. } m \text{ to dim. } n \} \]

\[ \sum \sum \text{tr}[(c(y))_{m,n}(\rho)] \leq 1, \forall m \in \mathbb{N}, \forall \rho \in D_m. \]

* Compare with

\[ PY = \{ c : Y \rightarrow 2 \} \]

\[ DY = \{ c : Y \rightarrow [0, 1] \mid \sum_{y \in Y} c(y) \leq 1 \} \]
The Quantum Branching

\[ QY = \left\{ c : Y \rightarrow \prod_{m, n \in \mathbb{N}} QO_{m, n} \right\} \quad \text{the trace condition} \]

\[ QO_{m, n} := \left\{ \text{quantum operations, from dim. } m \text{ to dim. } n \right\} \]

\[ \sum\sum_{y \in Y \ n \in \mathbb{N}} \text{tr}\left[ (c(y))_{m, n}(\rho) \right] \leq 1 , \quad \forall m \in \mathbb{N}, \forall \rho \in D_m. \]

Compare with

\[ PY = \left\{ c : Y \rightarrow 2 \right\} \]

\[ DY = \left\{ c : Y \rightarrow [0, 1] \left| \sum_{y \in Y} c(y) \leq 1 \right\} \right\]
The Quantum Branching Monad

\[ \mathcal{Q}Y = \left\{ c : Y \to \prod_{m,n \in \mathbb{N}} QO_{m,n} \mid \text{the trace condition} \right\} \]

\[ \sum_{y \in Y} \sum_{n \in \mathbb{N}} \text{tr}[(c(y))_{m,n}(\rho)] \leq 1 \]

\[ \forall m \in \mathbb{N}, \forall \rho \in D_m. \]

* Given \( x \in X, y \in Y, m \in \mathbb{N}, n \in \mathbb{N} \) determines a quantum operation

\[ \left( f(x)y \right)_{m,n} : D_m \to D_n \]

* Subject to the trace condition
The Quantum Branching Monad

\[ QY = \{ c : Y \to \prod_{m,n \in \mathbb{N}} QO_{m,n} \mid \text{the trace condition} \} \]

\[ \sum_{y \in Y} \sum_{n \in \mathbb{N}} \text{tr}[(c(y))_{m,n}(\rho)] \leq 1, \quad \forall m \in \mathbb{N}, \forall \rho \in D_m. \]

\[ X \xrightarrow{f} Y \text{ in } \mathcal{K}l(Q) \]

\[ X \xrightarrow{f} QY \text{ in Sets} \]

* Given \( x \in X, y \in Y, m \in \mathbb{N}, n \in \mathbb{N} \)

determines a quantum operation

\[ \left( f(x)(y) \right)_{m,n} : D_m \to D_n \]

* Subject to the trace condition

Any opr. on quantum data:

- combination of
  - preparation
  - unitary transf.
  - measurement
The Quantum Branching Monad

\[ \mathcal{Q}Y = \left\{ c : Y \rightarrow \prod_{m,n \in \mathbb{N}} \mathcal{Q}O_{m,n} \right\} \]

\[ \sum_{y \in Y} \sum_{n \in \mathbb{N}} \text{tr}[(c(y))_{m,n}(\rho)] \leq 1 , \]

\[ \forall m \in \mathbb{N}, \forall \rho \in D_m. \]

* Given \( x \in X, y \in Y, m \in \mathbb{N}, n \in \mathbb{N} \) determines a quantum operation \( (f(x)(y))_{m,n} \)

* trace cond.:
The Quantum Branching Monad

\[ QY = \left\{ c : Y \rightarrow \prod_{m,n \in \mathbb{N}} QO_{m,n} \mid \text{the trace condition} \right\} \]

\[ \sum_{y \in Y} \sum_{n \in \mathbb{N}} \text{tr}[(c(y))_{m,n}(\rho)] \leq 1, \quad \forall m \in \mathbb{N}, \forall \rho \in D_m. \]

* Given \( x \in X, y \in Y, m \in \mathbb{N}, n \in \mathbb{N} \)

determines a quantum operation \( (f(x)(y))_{m,n} \)

* trace cond.:
The Quantum Branching Monad

\[ QY = \{ c : Y \to \prod_{m,n \in \mathbb{N}} QO_{m,n} \mid \text{the trace condition} \} \]

\[ \sum_{y \in Y} \sum_{n \in \mathbb{N}} \text{tr}[(c(y))_{m,n}(\rho)] \leq 1, \quad \forall m \in \mathbb{N}, \forall \rho \in D_m. \]

Given \( x \in X, y \in Y, m \in \mathbb{N}, n \in \mathbb{N} \)
determines a quantum operation \( (f(x)(y))_{m,n} \)

\[ \begin{array}{c}
\text{entrance} \\
\text{exit} \\
\text{in dim.} \\
\text{out dim.}
\end{array} \]

\[ X \xrightarrow{f} Y \text{ in } \mathcal{Kl}(Q) \]

\[ X \xrightarrow{f} QY \text{ in Sets} \]

\( x \)
\( \rho \in D_m \)
\( \cdots \)
\( y \)
\( y' \)

\[ \begin{array}{c}
\text{in dim.} \\
\text{out dim.}
\end{array} \]
The Quantum Branching Monad

\[
\mathcal{Q}_Y = \left\{ c : Y \to \prod_{m,n \in \mathbb{N}} \mathcal{Q}_{O_{m,n}} \mid \text{the trace condition} \right\}
\]

\[
\sum_{y \in Y} \sum_{n \in \mathbb{N}} \text{tr} \left[ (c(y))_{m,n}(\rho) \right] \leq 1 , \quad \forall m \in \mathbb{N}, \forall \rho \in D_m.
\]

* Given \( x \in X, y \in Y, m \in \mathbb{N}, n \in \mathbb{N} \)
determines a quantum operation \( (f(x)(y))_{m,n} \)

* trace cond.:

\[ f : X \to Y \quad \text{in} \quad \mathcal{K}\ell(Q) \]
\[ X \xrightarrow{f} \mathcal{Q}_Y \quad \text{in} \quad \text{Sets} \]

\( \rho \in D_m \)

\( x \)

... 

\( y \)

... 

\( y' \)

measure (and others)
The Quantum Branching Monad

Given \( x \in X, y \in Y, m \in \mathbb{N}, n \in \mathbb{N} \),
determines a quantum operation \( (f(x)(y))_{m,n} \).

trace cond.:

\[
\mathcal{Q} Y = \left\{ c : Y \to \prod_{m,n \in \mathbb{N}} QO_{m,n} \mid \text{the trace condition} \right\}
\]

\[
\sum_{y \in Y} \sum_{n \in \mathbb{N}} \text{tr}[(c(y))_{m,n}(\rho)] \leq 1, \quad \forall m \in \mathbb{N}, \forall \rho \in D_m.
\]

\( \rho \in D_m \)

measure (and others)

for each \( n \)
The Quantum Branching Monad

Given \( x \in X, y \in Y, m \in \mathbb{N}, n \in \mathbb{N} \)

determines a quantum operation \( (f(x)(y))_{m,n} \)

\[
\sum_{y,n} \Pr\left(\begin{array}{c}
\text{Token led} \\
\text{to } y \\
\text{with dim. } n
\end{array}\right) \leq 1
\]

\[
\sum_{y \in Y} \sum_{n \in \mathbb{N}} \text{tr}[(c(y))_{m,n}(\rho)] \leq 1,
\]

\( \forall m \in \mathbb{N}, \forall \rho \in D_m \).

\[Q_Y = \left\{ c : Y \to \prod_{m,n \in \mathbb{N}} Q_0_{m,n} \mid \text{the trace condition} \right\}\]
"Quantum Data, Classical Control"

Quantum data

Illustration by N. Hoshino

Classical control

Hasuo (Tokyo)
"Quantum Data, Classical Control"

Illustration by N. Hoshino

Quantum data

\[ \frac{1}{\sqrt{2}} \]

Classical control

Hasuo (Tokyo)
“Quantum Data, Classical Control”

Quantum data

\[ \frac{1}{\sqrt{2}} + \frac{1}{\sqrt{2}} \]

Illustration by N. Hoshino

Classical control
Quantum Geometry of Interaction

\[ [M] = M \]

Hasuo (Tokyo)

Tuesday, October 9, 12
Quantum
Geometry of Interaction

Not just a token/particle, but quantum state!

\[
\begin{bmatrix}
M
\end{bmatrix} = \ldots
\]

1 2 3 4 ... (countably many)
Quantum Geometry of Interaction

\[ [M] = M \]

Not just a token/particle, but quantum state!
Quantum Geometry of Interaction

$\left[ M \right] = M$

"Quantum Data"

Not just a token/particle, but quantum state!
Quantum Geometry of Interaction

Not just a token/particle, but quantum state!

"Quantum Data"

"Classical Control"

(countably many)
Quantum Geometry of Interaction

\[ [M] = \]

“Quantum Data”

Not just a token/particle, but quantum state!

“Classical Control”

* “in which pipe”
* (measurement \(\rightarrow\) case-distinction) leads a token to different pipes

Tuesday, October 9, 12
End of the Story?

* No! All the technicalities are yet to come:
  * CPS-style interpretation (for partial measurement)
  * Result type: a final coalgebra in $\text{PER}_Q$
  * Admissible PERs for recursion
  * ...

* On the next occasion :-)
Results

* The monad $\mathcal{Q}$ qualifies as a “branching monad”

* The quantum GoI workflow leads to a linear category $\text{PER}_\mathcal{Q}$

* From which we construct an adequate denotational model for a quantum $\lambda$-calculus (a variant of Selinger & Valiron’s)
Three “Traces”

Coalgebraic Trace Semantics

Traced monoidal category

Quantum λ-calculus

Categorical GoI
[Abramsky, Haghverdi, Scott]

Measurements by tracing out matrices

Hasuo (Tokyo)

\[
\begin{align*}
F \text{beh}(c) : FX & \to FZ \\
c & \uparrow \\
X & \to Y \\
\text{beh}(c) & \uparrow \text{final}
\end{align*}
\]

Kl(B)
Conclusions & Future Work

* Coalgebraic technologies in interaction-based denotational semantics
  * GoI, games (AJM/HO), token machines, ...
  * Dynamic/operational stuff: not only in concurrency theory!

* Simplifying our model; lang. w/ “quantum store”
  * Ongoing w/ N. Hoshino, T. Roussel, C. Faggian
Conclusions & Future Work

* Coalgebraic technologies in interaction-based denotational semantics
  
  * GoI, games (AJM/HO), token machines, ...

* Dynamic/operational stuff:
  
  not only in concurrency theory!

* Simplifying our model; lang. w/ “quantum store”

* Ongoing w/ N. Hoshino, T. Roussel, C. Faggian