Semantics of Higher-Order Quantum Computation via Geometry of Interaction

In: Proc. Logic in Computer Science (LICS), June 2011

Ichiro Hasuo
University of Tokyo (JP)

Naohiko Hoshino
RIMS, Kyoto University (JP)
Contribution

Denotational semantics of a functional quantum programming language
Contribution

Denotational semantics of a functional quantum programming language

Linear $\lambda$-calculus + quantum primitives
Contribution

One of the first to cover the full features!

- !-modality for duplicable data
- recursion

Denotational semantics of a functional quantum programming language

Linear λ-calculus + quantum primitives
Contributions

Denotational semantics of a functional quantum programming language

One of the first to cover the full features!
- !-modality for duplicable data
- recursion

Adequacy

Linear $\lambda$-calculus + quantum primitives

Hasuo (Tokyo)
Denotational semantics of a functional quantum programming language

One of the first to cover the full features!

- !-modality for duplicable data
- recursion

Adequacy

Linear \( \lambda \)-calculus + quantum primitives

... via GoI (Geometry of Interaction)

Hasuo (Tokyo)
Part 1

Functional QPL: Some Contexts
# Quantum Programming Language

## Classical vs. Quantum

<table>
<thead>
<tr>
<th>Classical</th>
<th>Quantum</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Boolean) circuit</td>
<td>Quantum circuit</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Programming language</th>
</tr>
</thead>
<tbody>
<tr>
<td>int i, j;</td>
</tr>
<tr>
<td>int factorial(int k)</td>
</tr>
<tr>
<td>{</td>
</tr>
<tr>
<td>j = 1;</td>
</tr>
<tr>
<td>for (i = 1; i &lt;= k; i++)</td>
</tr>
<tr>
<td>j = j * i;</td>
</tr>
<tr>
<td>return j;</td>
</tr>
<tr>
<td>}</td>
</tr>
</tbody>
</table>

Hasuo (Tokyo)
# Quantum Programming Language

<table>
<thead>
<tr>
<th>Classical</th>
<th>Quantum Programming Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Boolean) circuit</td>
<td>Quantum circuit</td>
</tr>
<tr>
<td><img src="Null-Lobur" alt="Classical circuit" /></td>
<td><img src="beachhandball.es" alt="Quantum circuit" /></td>
</tr>
<tr>
<td>Programming language</td>
<td>Quantum programming language</td>
</tr>
<tr>
<td>int i,j; int factorial(int k) { j=1; for (i=1; i&lt;=k; i++) j=j*i; return j; }</td>
<td>autocall</td>
</tr>
</tbody>
</table>
# Quantum Programming Language

<table>
<thead>
<tr>
<th>Classical</th>
<th>Quantum</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Boolean) circuit</td>
<td>Quantum circuit</td>
</tr>
<tr>
<td>Programming language</td>
<td>Programming language</td>
</tr>
</tbody>
</table>

```c
int i, j;
int factorial(int k)
{
    j = 1;
    for (i = 1; i <= k; i++)
    {
        j = j * i;
    }
    return j;
}
```

- For discovery of **algorithms**
- For **reasoning**, **verification**

Monday, November 7, 2011
Functional Quantum Programming Language
Functional Quantum Programming Language

* A real man’s programming style

Hasuo (Tokyo)
Functional Quantum Programming Language

* A real man’s programming style
* Heavily used in the financial sector
* ...

ICFP’11 Sponsors (Tokyo, Sep 2011)

Monday, November 7, 2011
**Functional Quantum Programming Language**

- A real man’s programming style
- Heavily used in the financial sector
- ...  

- Mathematically nice and clean
- Aids semantical study
- Transfer from classical to quantum
Functional QPL: Syntax

* Linear $\lambda$-calculus + quantum primitives [van Tonder, Selinger, Valiron, ...]

* Linearity for no-cloning
  * "Input can be used only once"
  * Not allowed/typable:

* Duplicable (classical) data: by the $!$-modality
Functional QPL:
Syntax

* Linear $\lambda$-calculus + quantum primitives
  [van Tonder, Selinger, Valiron, ...]

* Linearity for no-cloning
  “Input can be used only once”

* Not allowed/typable:
  Duplicable (classical) data: by the $!$-modality
**Functional QPL: Syntax**

- Linear $\lambda$-calculus + quantum primitives [van Tonder, Selinger, Valiron, ...]
- Linearity for **no-cloning**
  - "Input can be used only once"
  - Not allowed/typable: $\lambda x. \langle \text{meas } x, \text{meas } x \rangle$
Functional QPL: Syntax

* Linear $\lambda$-calculus
  + quantum primitives

* Linearity for no-cloning

  "Input can be used only once"

* Not allowed/typable:

  $\lambda x. \langle \text{meas } x, \text{meas } x \rangle$

[van Tonder, Selinger, Valiron, ...]

Preparation/Unitary transformation/Measurement

Hasuo (Tokyo)
Functional QPL: Syntax

- Linear $\lambda$-calculus + quantum primitives
- Linearity for no-cloning
  - “Input can be used only once”
  - Not allowed/typable:
    $\lambda x. \langle \text{meas } x, \text{meas } x \rangle$
- Duplicable (classical) data: by the $!$-modality
  $\vdash \text{tt : } !\text{bit}$

[van Tonder, Selinger, Valiron, ...]
Functional QPL: Syntax

* Linear λ-calculus + quantum primitives

* Linearity for no-cloning

* “Input can be used only once”

* Not allowed/typable:

  \[ \lambda x. \langle \text{meas } x, \text{meas } x \rangle \]

* Duplicable (classical) data: by the \( ! \)-modality

  \[ \vdash \text{tt} : !\text{bit} \]

  “arbitrary many copies”
Functional QPL: Semantics
Functional QPL:
Semantics

Semantics = mathematical model
Functional QPL: Semantics

* Semantics = mathematical model

* Operational semantics: [Selinger & Valiron, ’09]

  “Quantum closure,”
  reduction with probabilistic branching

  \[
  [\alpha|Q_0\rangle + \beta|Q_1\rangle, |x_1 \ldots x_n\rangle, \text{meas } x_i] \rightarrow_{|\alpha|^2} [Q_0, |x_1 \ldots x_n\rangle, 0] \\
  [\alpha|Q_0\rangle + \beta|Q_1\rangle, |x_1 \ldots x_n\rangle, \text{meas } x_i] \rightarrow_{|\beta|^2} [Q_1, |x_1 \ldots x_n\rangle, 1]
  \]

* Allows to identify linear logic \(\otimes\) and quantum \(\otimes\)
  (feature of the Selinger–Valiron language; not in ours)
Functional QPL: Semantics

$[\mathcal{M}]$
Functional QPL: Semantics

* Denotational semantics

* $[M]$: a function, or an arrow of a category
Functional QPL: Semantics

* Denotational semantics

* $\llbracket M \rrbracket$: a function, or an arrow of a category

* Compositionality: $\llbracket MN \rrbracket = \llbracket M \rrbracket \circ \llbracket N \rrbracket$
Functional QPL: Semantics

* Denotational semantics

* \([M]\): a function, or an arrow of a category

* Compositionality: \([MN] = [M] \circ [N]\)

* Linear category: [Benton & Wadler, Bierman]
  (axioms for) a categorical model of linear \(\lambda\)-calculus

Defn.
A linear category \((\mathcal{C}, \otimes, I, \multimap, !)\) is a sym. monoidal closed cat. with a linear exponential comonad !.
Functional QPL: Semantics

- **Denotational semantics**

- **Compositionality**: $[MN] = [M] \circ [N]$

- **Linear category**: [Benton & Wadler, Bierman]
  (axioms for) a categorical model of linear $\lambda$-calculus

**Defn.**
A linear category $\mathbb{C}$ is a sym. monoidal closed cat. with a linear exponential comonad $!$.

**For functional QPL? Is** **Hilb** (or alike) a linear cat.?
Functional QPL: Semantics

* Hilb (or alike) is not a linear category

* **Challenge**: coexistence of quantum and classical data

* Only partial results
  * [Selinger & Valiron, ’08]: for strictly linear fragment (w/o !)
Functional QPL: Semantics

* Hilb (or alike) is **not** a linear category

monoidal closed str. \((\mathbb{C}, \otimes, I, \to)\)

! (for duplicable data)

* **Challenge**: coexistence of quantum and classical data

* Only partial results

  * [Selinger & Valiron, ’08]:
    for strictly linear fragment (w/o !)
Functional QPL: Semantics

* Hilb (or alike) is not a linear category

monoidal closed str. \((\mathbb{C}, \otimes, I, \sim)\)

\[ \text{duality } V \cong V^\perp \]

* Challenge: coexistence of quantum and classical data

* Only partial results

  * [Selinger & Valiron, ‘08]:
    for strictly linear fragment (w/o !)
Functional QPL: Semantics

* Hilb (or alike) is not a linear category

\[
\text{monoidal closed str. } (\mathbb{C}, \otimes, I, \rightarrow)
\]

! (for duplicable data)

\[ V \cong V^\perp \]

\text{finite dim.}

* Challenge: coexistence of quantum and classical data

* Only partial results

* [Selinger & Valiron, ‘08]:
  for strictly linear fragment (w/o !)
Functional QPL: Semantics

- Hilb (or alike) is not a linear category
- Challenge: coexistence of quantum and classical data
- Only partial results
  - [Selinger & Valiron, ’08]: for strictly linear fragment (w/o ! )

\[
\text{duality } V \cong V^\perp
\]

\[
\text{monoidal closed str. } (\mathbb{C}, \otimes, I, \rightarrow)
\]

\[\text{! (for duplicable data)}\]

\[\text{infinite dim.}\]

\[\text{finite dim.}\]
Functional QPL: Semantics

- Hilb (or alike) is **not** a linear category

  monoidal closed str. \((\mathbb{C}, \otimes, I, \neg)\)

  **duality** \(V \cong V^\perp\)

  finite dim.

  infinite dim.

- **Challenge**: coexistence of quantum and classical data
**Functional QPL: Semantics**

* Hilb (or alike) is **not** a linear category

  monoidal closed str. \((\mathbb{C}, \otimes, I, \sim)\)

  \[ \text{duality } V \cong V^\perp \]

  \[ \text{finite dim. } \]

  \[ \text{infinite dim. } \]

* **Challenge:** coexistence of quantum and classical data

* Only partial results

* [Selinger & Valiron, ’08]:
  for strictly linear fragment (w/o ! )

Hasuo (Tokyo)

Monday, November 7, 2011
“Quantum Data, Classical Control”

Quantum data

Classical control

Illustration by N. Hoshino
“Quantum Data, Classical Control”

Quantum data

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

Illustration by N. Hoshino
"Quantum Data, Classical Control"

Quantum data

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

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

Classical control

Illustration by N. Hoshino

Monday, November 7, 2011
What We Do

* **GoI (Geometry of Interaction)** [Girard ’89]
  
  An “implementation” of **classical control**

\[
\text{tr}(f) = f_{XY} \sqcup \left( \bigsqcup_{n \in \mathbb{N}} f_{ZY} \circ (f_{ZZ})^n \circ f_{XZ} \right)
\]
What We Do

* GoI (Geometry of Interaction) [Girard ’89]
  An “implementation” of classical control

\[ \text{tr}(f) = f_{XY} \sqcup \left( \bigsqcup_{n \in \mathbb{N}} f_{ZY} \circ (f_{ZZ})^n \circ f_{XZ} \right) \]
What We Do

* **GoI (Geometry of Interaction)** [Girard ’89]
An “implementation” of classical control

* **Categorical GoI** [Abramsky, Haghverdi, Scott ’02]
Its categorical axiomatics
What We Do

* **GoI (Geometry of Interaction)** [Girard ’89]
  An “implementation” of classical control

* **Categorical GoI** [Abramsky, Haghverdi, Scott ’02]
  Its categorical axiomatics

* We add a quantum layer to GoI
  * “Quantum data, classical control”
  * Used: theory of coalgebra
    [Hasuo, Jacobs, Sokolova ’07] [Jacobs ’10]
Part 2

The Categorical GoI Workflow
GoI: Geometry of Interaction

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

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

* J.-Y. Girard, at Logic Colloquium ‘88

* **But** I’m no linear logician!
GoI: Geometry of Interaction

* J.-Y. Girard, at Logic Colloquium ’88

* But I’m no linear logician!

* 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”

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

... (countably many)
The GoI Animation

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

= “piping”

1 2 3 4 ...

(countably many)
The GoI Animation

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

= “piping”

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

... (countably many)

Monday, November 7, 2011
\[ \text{The GoI Animation} \]

\[
[M] = (\mathbb{N} \rightarrow \mathbb{N}, \text{a partial function}) = \text{“piping”} \\
1 \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \\
\quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \ Quad
The GoI Animation

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

= “piping”

\[ \downarrow \downarrow \downarrow \downarrow \downarrow \]

1 2 3 4 ...

(countably many)
The GoI Animation

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

= “piping”

\[ 1 \quad 2 \quad 3 \quad 4 \quad \ldots \]

(countably many)
The GoI Animation

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

= “piping”

1 2 3 4 ...

(countably many)
The GoI Animation

* Function application \([MN]\)

* by "parallel composition + hiding"
\[ M N \] = \begin{bmatrix} M \end{bmatrix} \begin{bmatrix} N \end{bmatrix}
\[ MN \] = [M] [N]
$[MN] = [M] [N]$
\[ [M N] = [M] \quad \text{and} \quad [N] \]
\[ M N \] = \[[M]\] \quad \cdots \quad \cdots \quad \cdots \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quarter
$[MN] = [M] \parallel [N]$  

"parallel composition + hiding" (cf. games)
\[ M \equiv \lambda x. x + 1 \quad N \equiv 2 \]
\[ M \equiv \lambda x. 1 \quad N \equiv 2 \]
\[ M \equiv \lambda f. f1 \quad N \equiv \lambda x. (x + 1) \]
\[ M N \]

\[ = \]

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

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

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

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

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

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

\[ = \]

\[ \boxed{M} \]

\[ \boxed{N} \]

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

\[ = \]

\[ [M] \]

\[ [N] \]

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

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

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

Monday, November 7, 2011
\[ [MN] = M = \lambda x. x + 1 \quad N = 2 \]
\[ \quad \rightarrow M = \lambda x. 1 \quad N = 2 \]
\[ \quad \rightarrow M = \lambda f. f1 \quad N = \lambda x. (x + 1) \]
\[ MN \] =

\[ [M] \]

\[ [N] \]

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

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

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

Monday, November 7, 2011
\[ [MN] = [M] \oplus [N] \]

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

\[ = \]

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

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

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

Hasuo (Tokyo)

Monday, November 7, 2011
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

Monday, November 7, 2011

Hasuo (Tokyo)
The Categorical GoI Workflow

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

Categorical GoI [AHS02]

Linear combinatory algebra

Realizability

Linear category

\[ \begin{array}{ccc}
A & f & B \\
B & C & B \\
\end{array} \quad \begin{array}{ccc}
A & \text{tr} & B \\
\text{tr}(f) & C & \text{tr}(f) \\
\end{array} \]

\[ \text{Model of typed calculus} \]

Hasuo (Tokyo)

Applicative str. + combinators

Model of untyped calculus

Monday, November 7, 2011
The Categorical GoI Workflow

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

Categorical GoI [AHS02]

Linear combinatory algebra

Realizability

Linear category

- Applicative str. + combinators
- Model of untyped calculus
- PER, ω-set, assembly, ...
- “Programming in untyped λ”

Model of typed calculus
The Categorical GoI Workflow

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

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)
  
  $\cdot : A^2 \to A$

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

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

  $Bxyz = x(yz)$ Composition, Cut
  $Cxyz = (xz)y$ Exchange
  $lx = x$ Identity
  $Kxy = x$ Weakening
  $Wxy = x!y!y$ Contraction
  $D!x = x$ Dereliction
  $\delta!x = !!x$ Comultiplication
  $F!x!y = !(xy)$ Monoidal functoriality

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

- $Bxyz = x(yz)$ Composition, Cut
- $Cxyz = (xz)y$ Exchange
- $Ix = x$ Identity
- $Kxy = x$ Weakening
- $Wxy = xxy$ Contraction
- $D!x = x$ Dereliction
- $\delta!x = !x$ Comultiplication
- $F!x = ! (xy)$ Monoidal functoriality

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

* Model of untyped linear $\lambda$
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

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

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

Model of untyped linear $\lambda$

- $a \in A \approx$ closed linear $\lambda$-term

What we want (outcome)
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

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

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

What we want (outcome)

- Model of untyped linear $\lambda$
- $a \in A \approx$ closed linear $\lambda$-term
- No $S$ or $K$ (linear!)
**Linear Combinatory Algebra (LCA)**

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:

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

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

---

What we want (outcome):

- Model of untyped linear $\lambda$
- $a \in A \approx$ closed linear $\lambda$-term
- No $S$ or $K$ (linear!)
- Combinatory completeness: e.g.

  $\lambda xyz. zxy$

  designates an elem. of $A$

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).
  
  
  
  \[
  \begin{align*}
  e & : FF \otimes F \to e' & \text{Comultiplication} \\
  d & : \text{id} \otimes F \to d' & \text{Dereliction} \\
  c & : F \otimes F \to c' & \text{Contraction} \\
  w & : K_I \otimes F \to w' & \text{Weakening}
  \end{align*}
  \]

  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.
  
  \[
  \begin{align*}
  j & : U \otimes U \otimes U \to k \\
  I & \otimes U \\
  u & : FU \otimes U \to v
  \end{align*}
  \]
**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 \triangleright F : e' \quad \text{Comultiplication}
  
  d : \text{id} \triangleright F : d' \quad \text{Dereliction}
  
  c : F \otimes F \triangleright F : c' \quad \text{Contraction}
  
  w : K_I \triangleright F : w' \quad \text{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.
  
  \[
  j : U \otimes U \triangleright U : k \\
  I \triangleright U \\
  u : FU \triangleright U : v
  \]

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

**String diagrams**
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).
  - \(e : F F \triangleleft F : e'\) 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 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).
  
  - $e : FF \otimes F \to e'$: Comultiplication
  - $d : \text{id} \otimes F \to d'$: Dereliction
  - $c : F \otimes F \otimes F \to c'$: Contraction
  - $w : K_I \otimes F \to w'$: 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.
  
  - $j : U \otimes U \otimes U \to k$
  - $I \otimes U$
  - $u : FU \otimes U \to v$

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

**String diagrams**

\[
A \xrightarrow{f} B \quad B \xrightarrow{g} C
\]

$A \xrightarrow{g \circ f} C$

\[
A \otimes C \xrightarrow{f \otimes g} B \otimes D
\]

$h \circ (f \otimes g)$

Monday, November 7, 2011
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).
  - $e : FF < F : e'$ 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.
  - $j : U \otimes U < U : k$
  - $I < U$
  - $u : FU < U : v$

Traced monoidal category

“feedback”

\[
\begin{align*}
A \otimes C & \rightarrow^f B \otimes C \\
\downarrow^{\text{tr}(f)} & \downarrow^\rightarrow \\
A & \rightarrow \text{tr}(f) B
\end{align*}
\]

that is

\[
\begin{array}{c}
A \\
\downarrow^f \\
B
\end{array}
\rightarrow
\begin{array}{c}
A \\
\downarrow^\text{tr}(f) \\
B
\end{array}
\]
String Diagram vs. “Pipe Diagram”

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

* Category Pfn of partial functions

* Obj. A set $X$

* Arr. A partial function

\[
\begin{array}{c}
X ightarrow Y \text{ in Pfn} \\
X \rightarrow Y, \text{ partial function}
\end{array}
\]
Traced Sym. Monoidal Category
(Pfn, +, 0)

* Category Pfn of partial functions

* Obj. A set $X$

* Arr. A partial function

$X \rightarrow Y$ in Pfn

$X \rightarrow Y$, partial function

* is traced symmetric monoidal
Traced Sym. Monoidal Category

\((Pfn, +, 0)\)

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

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


Traced Sym. Monoidal Category

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

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

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

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

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

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

Monday, November 7, 2011
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?

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

Similar for \(f_{xz}, f_{zy}, f_{zz}\)

Trace operator:

\[
\begin{aligned}
X & \quad f \\
Y & \quad Z
\end{aligned}
\]
Traced Sym. Monoidal Category
(Pfn, +, 0)

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

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:

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

\[(Pfn, +, 0)\]

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

How?

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

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

* Execution formula (Girard)

* Partiality is essential (infinite loop)

\[\text{tr}(f) = \bigcup f_{XY} \sqcup \left( \bigoplus_{n \in \mathbb{N}} f_{ZY} \circ (f_{ZZ})^n \circ f_{XZ} \right)\]
**GoI situation**

* 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).
    - \(e : FF \triangleleft F : e'\) \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}

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.
    - \(j : U \otimes U \triangleleft U : k\)
    - \(I \triangleleft U\)
    - \(u : FU \triangleleft U : v\)

* Where one can “feedback”

* Why for GoI?
\[ MN \] = [M] \cdots [N] \]
\[ [MN] = [M] \] in string diagram

Monday, November 7, 2011
**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);
- \(F : \mathcal{C} \rightarrow \mathcal{C}\) 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.
  - \(j : U \otimes U < U : k\)
  - \(I < U\)
  - \(u : FU < U : v\)

**Traced sym. monoidal cat.**

- Where one can “feedback”

**Why for GoI?**

**Leading example:** Pfn

---

Monday, November 7, 2011
**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 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

\[
\text{id} \quad X \quad Y
\]

\[
f \quad g
\]

such that \(g \circ f = \text{id}_X\).

\[\star\] **Functor \(F\)**

\[\star\] **For obtaining \(! : A \to A\)**
**GoI situation**

* The reflexive object $U$

* Retr. $U \otimes U \overset{j}{\leftrightarrow} U \overset{k}{\Leftarrow} U$

---

**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).
  
  $e : FF \triangleleft F : e'$  
  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 $I$;

- $U \in \mathcal{C}$ 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).
  
  $e : FF \triangleleft F : e'$ 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 retraction.
  
  $j : U \otimes U \triangleleft U : k$
  $I \triangleleft U$
  $u : FU \triangleleft U : v$

**The reflexive object $U$**

**Retr.** $U \otimes U \xleftarrow{j} U \xrightarrow{k}$ with

$\begin{align*}
 j, \quad k \\
 \downarrow \quad \downarrow \\
 j, \quad k
\end{align*}$

$= \text{id}$

Hasuo (Tokyo)
**GoI situation**

* The reflexive object \( U \)

* Why for GoI?

\[ \text{Defn. (GoI situation [AHS02])} \]

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
  - \( d : \text{id} < F : d' \)
  - \( c : F \otimes F < F : c' \) Contraction
  - \( w : K_I < F : w' \)

Here \( K_I \) is the constant functor.

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

* 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 $(\mathbb{C}, F, U)$ 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).
  
  $e : FF \triangleleft F : e' \quad \text{Comultiplication}$
  $d : \text{id} \triangleleft F : d' \quad \text{Dereliction}$
  $c : F \otimes F \triangleleft F : c' \quad \text{Contraction}$
  $w : K_I \triangleleft F : w' \quad \text{Weakening}$

Here $K_I$ is the constant functor.

- $U \in \mathbb{C}$ is an object (called reflexive object), equipped with the following retractions.
  
  $j : U \otimes U \triangleleft U : k$
  $I \triangleleft U$
  $u : FU \triangleleft U : v$
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).
  - \(e : FF \triangleleft F \triangleleft e'\) Comultiplication
  - \(d : \text{id} \triangleleft F \triangleleft d'\)
  - \(c : F \otimes F \triangleleft F \triangleleft c'\) Contraction
  - \(w : K_I \triangleleft F \triangleleft w'\) Weakening

Here \(K_I\) is the constant functor.

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

**Why for GoI?**

**Example in Pfn:**

\[ N \in \text{Pfn}, \text{ with } N + N \cong N, \]
\[ N \cdot N \cong N \]
**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).
  - \(e : FF \otimes F \to F : e'\) 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.
  - \(j : U \otimes U \to U : k\)
  - \(I \otimes U \to U : v\)

**Example:**

\[(\text{Pfn}, N \cdot \_, N)\]
Defn. (GoI situation [AHS02])

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

- \(C = (\mathcal{C}, \otimes, I)\) 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 < F : e' & \text{Comultiplication} \\
  d & : \text{id} < F : d' & \text{Dereliction} \\
  c & : F \otimes F < F : c' & \text{Contraction} \\
  w & : K_I < F : w' & \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 < U : k \\
  I & < U \\
  u & : FU < U : v
  \end{align*}

Example:

\((\text{Pfn}, N \cdot \_ \_ N)\)
**Example:**

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

**Example:**

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

Example:

\[ (\text{Pfn}, N \cdot \_, N) \]
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 \((\mathcal{C}, \mathbf{F}, \mathbf{U})\), the homset
\[ \mathcal{C}(U, U) \]
carries a canonical LCA structure.
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, ...
**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, ...

\[
\begin{align*}
g \cdot f &:= \text{tr}((U \otimes f) \circ k \circ g \circ j) \\
&= \begin{array}{c}
g \\
n \\
= \\
g \\
f \\
f \\
= \\
f \\
g \\
g \\
g \\
g \end{array}
\end{align*}
\]
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

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

Figure 7: Composition Combinator B

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

* Combinator $Bxyz = x(yz)$
Hasuo (Tokyo)

Monday, November 7, 2011

Categorical GoI:

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

\[ B_{x y z} = x(yz) \]
Hasuo (Tokyo)

Categorical GoI:

Constr. of an LCA

Combinator

\[ Bxyz = x(yz) \]
Categorical GoI:

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

* Combinator  $Bxyz = x(yz)$

Figure 7: Composition Combinator B

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

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

Figure 7: Composition Combinator B

Nice dynamic interpretation of (linear) computation!!
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).
  - $e : FF \otimes F : e'$ Comultiplication
  - $d : \text{Id} \otimes F : d'$ Dereliction
  - $c : F \otimes F \otimes F : c'$ Contraction
  - $w : K_I \otimes F : w'$ 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.
  - $j : U \otimes U \otimes U : k$
  - $I \otimes U
  - $u : FU \otimes U : v$

**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 \texttt{Pfn} \cite{AHS02}

\begin{itemize}
  \item Strategy: find a TSMC!
  \item "Wave-style" examples
    \begin{itemize}
      \item \(\otimes\) is Cartesian product(-like)
      \item in which case,
      \end{itemize}
  \end{itemize}

\textcolor{red}{\texttt{trace} \approx \texttt{fixed point operator}} \cite{Hasegawa/Hyland}

\begin{itemize}
  \item An example: \((\omega\texttt{-Cpo}, \times, 1), (\_)^\mathbb{N}, A^\mathbb{N}\)
  \item (... less of a dynamic flavor)
\end{itemize}
Why Categorical Generalization?: Examples Other Than Pfn [AHS02]

* “Particle-style” examples
  * Obj. \( X \in C \) is set-like; \( \otimes \) is coproduct-like
  * The GoI animation is valid

* Examples:
  * Partial functions
  \[ ( (Pfn, +, 0), \mathbb{N} \cdot \_ , \mathbb{N} ) \]
  * Binary relations
  \[ ( (Rel, +, 0), \mathbb{N} \cdot \_ , \mathbb{N} ) \]
  * “Discrete stochastic relations”
  \[ ( (DSRel, +, 0), \mathbb{N} \cdot \_ , \mathbb{N} ) \]
Why Categorical Generalization?:
Examples Other Than Pfn [AHS02]

* **Pfn (partial functions)**

\[
\begin{align*}
X &\to Y \text{ in Pfn} \\
X &\to Y, \text{ partial function} \\
X &\to \mathcal{L}Y \text{ in Sets}
\end{align*}
\]
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 \( \mathcal{D}Y = \{ d : Y \to [0, 1] | \sum_y d(y) \leq 1 \} \)
Why Categorical Generalization? Examples Other Than Pfn [AHS02]

* **Pfn** (partial functions)

  \[
  X \to Y \text{ in Pfn} \\
  X \to Y, \text{ partial function} \\
  X \to \mathcal{L}Y \text{ in Sets}
  \]

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

* **Rel** (relations)

  \[
  X \to Y \text{ in Rel} \\
  R \subseteq X \times Y, \text{ relation} \\
  X \to \mathcal{P}Y \text{ in Sets}
  \]

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

* **DSRel**

  \[
  X \to Y \text{ in DSRel} \\
  X \to \mathcal{D}Y \text{ in Sets}
  \]

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

Examples Other Than Pfn

* **Pfn** (partial functions)

  \[
  X \to Y \text{ in } \text{Pfn} \quad \frac{X \to Y, \text{ partial function}}{X \to \mathcal{L}Y \text{ in } \text{Sets}} \quad \text{where } \mathcal{L}Y = \{\bot\} + Y
  \]

  (Potential) non-termination

* **Rel** (relations)

  \[
  X \to Y \text{ in } \text{Rel} \quad \frac{R \subseteq X \times Y, \text{ relation}}{X \to \mathcal{P}Y \text{ in } \text{Sets}} \quad \text{where } \mathcal{P} \text{ is the powerset monad}
  \]

  Non-determinism

* **DSRel**

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

  Probabilistic branching

---

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

Hasuo (Tokyo)
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)

Monday, November 7, 2011
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

- **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
Why Categorical Generalization?:
Examples Other Than Pfn \[AHS02\]

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

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

- **DSRel**
  \[
  \frac{X \to Y \text{ in DSRel}}{X \to \mathcal{D}Y \text{ in Sets}} \quad \text{where } \mathcal{D}Y = \{d : Y \to [0, 1] \mid \sum_y d(y) \leq 1\}
  \]
Why Categorical Generalization?: Examples Other Than Pfn [AHS02]

* **Pfn (partial functions)**

\[
\frac{X \rightarrow Y \text{ in Pfn}}{X \rightarrow Y, \text{ partial function}}
\]

\[
\frac{X \rightarrow \mathcal{LY} \text{ in Sets}}{\mathcal{LY} = \{ \bot \} + Y}
\]

* **Rel (relations)**

\[
\frac{X \rightarrow Y \text{ in Rel}}{R \subseteq X \times Y, \text{ relation}}
\]

\[
\frac{X \rightarrow \mathcal{PY} \text{ in Sets}}{\mathcal{PY} \text{ is the powerset monad}}
\]

* **DSRel**

\[
\frac{X \rightarrow Y \text{ in DSRel}}{X \rightarrow \mathcal{DY} \text{ in Sets}}
\]

\[
\mathcal{DY} = \{ d : Y \rightarrow [0, 1] \mid \sum_y d(y) \leq 1 \}
\]

Essential to have subdistribution, for infinite loops
The Coauthor

Naohiko Hoshino

DSc (Kyoto, 2011)

Supervisor: Masahito “Hassei” Hasegawa

Currently at RIMS, Kyoto U.

http://www.kurims.kyoto-u.ac.jp/~naophiko/
A Coalgebraic View

- Theory of coalgebra = Categorical theory of state-based dynamic systems (LTS, automaton, Markov chain, ...)

- In [Hasuo, Jacobs, Sokolova ’07]:
  - Coalgebras in a Kleisli category $Kl(B)$
  - Generic theory of “trace semantics”
Why Categorical Generalization?
Examples Other Than Pfn

* **Pfn** (partial functions)

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

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

(Potential) non-termination
Non-determinism
Probabilistic branching

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

Monday, November 7, 2011
Why Categorial Generalization?

Examples Other Than Pfn

**Pfn** (partial functions)

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

\[ X \to Y, \text{ partial function} \]

\[ X \to \mathcal{LY} \text{ in Sets} \]

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

---

**Rel** (relations)

\[ X \to Y \text{ in Rel} \]

\[ R \subseteq X \times Y, \text{ relation} \]

\[ X \to \mathcal{PY} \text{ in Sets} \]

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

---

**DSRel**

\[ X \to Y \text{ in DSRel} \]

\[ X \to \mathcal{DY} \text{ in Sets} \]

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

---

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

(Potential) non-termination

Non-determinism

Probabilistic branching
Branching Monad: Source of Particle-Style GoI Situations

**Thm.** ([Jacobs,CMCS10])
Given a “branching monad” \( B \) on \( \text{Sets} \), the monoidal category
\[(K\ell(B), +, 0)\]
is

- a *unique decomposition category* [Haghverdi,PhD00], hence is
- a traced symmetric monoidal category.

**Cor.**
\[( (K\ell(B), +, 0), \mathbb{N} \cdot _-, \mathbb{N} ) \] is a GoI situation.
Branching Monad: Source of Particle-Style GoI Situations

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

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

is

- a unique decomposition category
  [Haghverdi,PhD00], hence is
- a traced symmetric monoidal category.

**Cor.**

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

(Roughly) monads in [Hasuo, Jacobs, Sokolova ’07]

- $\mathcal{K}\ell(B)$ is $\text{Cpo}_\perp$-enriched
- like $\mathcal{L}$, $\mathcal{P}$, $\mathcal{D}$
Thm. ([Jacobs, CMCS10])

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

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

is

• a unique decomposition category [Haghverdi, PhD00], hence is

• a traced symmetric monoidal category.

Cor.

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

(Roughly) monads in [Hasuo, Jacobs, Sokolova ’07]

• $\mathcal{Kl}(B)$ is $\text{Cpo}_\bot$-enriched

• like $\mathcal{L}$, $\mathcal{P}$, $\mathcal{D}$

Particle-style: trace via the execution formula

$$\text{tr}(f) = f_{xy} \sqcup \left( \bigsqcup_{n \in \mathbb{N}} f_{yz} \circ (f_{zz})^n \circ f_{xz} \right)$$

Hasuo (Tokyo)
The Categorical GoI Workflow

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

  ➜

  Categorical GoI [AHS02]

  ➜

  Linear combinatory algebra

  ➜

  Realizability

  ➜

  Linear category

Hasuo (Tokyo)

Monday, November 7, 2011
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 $B$

$\downarrow$

Coalgebraic trace semantics

Traced monoidal category $\mathcal{C}$

$+ \text{ other constructs } \rightarrow \text{“GoI situation” [AHS02]}

$\downarrow$

Categorical GoI [AHS02]

Linear combinatory algebra

$\downarrow$

Realizability

Linear category

Hasuo (Tokyo)

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)

Monday, November 7, 2011
The Categorical GoI Workflow

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

- Model of fancy language
  - Fancy LCA
  - Fancy TSMC

Monday, November 7, 2011
The Categorical GoI Workflow

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

- Model of fancy language
- Fancy LCA
- Fancy TSMC
- Fancy monad

Monday, November 7, 2011
What is Fancy, Nowadays?
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, Hasuo ICALP'11]
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, Hasuo ICALP’11]

* Quantum?
  * Yes this worked!
Future Directions

- GoI 2: Non-converging algs (untyped \(J\)-calc / PCF)
  - uses more topological info on operati alg

- GoI 3: uses additives & additive proof nets

- GoI 4 (last month): von Neumann algebras:
  \[ \text{Ex}(f, z) \quad f \quad \text{arb (not coming from proof)} \]

- Quantum GoI ?
Part 3

Phil Scott.
Page 47/47

Future Directions

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

- GoI 3: uses additives & additive
  proof nets —

- GoI 4 (last month): von Neumann
  algebras: \( \text{Ex}(f, z) \) for
  \( a \land b \) (not necessarily coming from proof)

Quantum GoI?
The Categorical GoI Workflow

1. Linear combinatory algebra
2. Realizability
3. Linear category
4. Categorical GoI [AHS02]
5. Traced monoidal category $\mathbb{C}$
   + other constructs $\Rightarrow$ “GoI situation” [AHS02]
6. Branching monad $B$
7. Coalgebraic trace semantics
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

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

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

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

\[ QO_{m,n} := \begin{cases} \text{quantum operations,} \\ \text{from dim. } m \text{ to dim. } n \end{cases} \]
The Quantum Branching Monad

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

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

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

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

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

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

\[ \text{the trace condition} \]

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

* 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

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

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

the trace condition

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

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

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

* Compare with

Hasuo (Tokyo)

Monday, November 7, 2011
The Quantum Branching

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

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

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

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

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

* Compare with
The Quantum Branching Monad

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

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

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

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

* 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
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

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

Any opr. on quantum data:
combination of
- preparation
- unitary transf.
- measurement
The Quantum Branching Monad

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

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

---

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

\[ X \xrightarrow{f} \mathcal{Q}Y \text{ in } \text{Sets} \]

* 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

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

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

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

* trace cond.:
The Quantum Branching Monad

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

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

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

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

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

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

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

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

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

\begin{itemize}
  \item Given \( x \in X, y \in Y, m \in \mathbb{N}, n \in \mathbb{N} \)
  \item determines a quantum operation \( (f(x)(y))_{m,n} \)
  \item trace cond.:
\end{itemize}

\( \rho \in D_m \) 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 $(f(x)(y))_{m,n}$

trace cond.:

$$\sum_{y,n} \Pr \left( \text{Token led to } y \text{ with dim. } n \right) \leq 1$$

$$Q_Y = \left\{ c : Y \to \prod_{m,n \in \mathbb{N}} QO_{m,n} \mid \text{the trace condition} \right\}$$

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

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

$\rho \in D_m$

$X \xrightarrow{f} Y$ in $\mathcal{Kl}(Q)$
$X \xrightarrow{f} QY$ in $\text{Sets}$

measure (and others)

entrance
exit
in dim.
out dim.

$X \xrightarrow{f} Y$ in $\mathcal{Kl}(Q)$
$X \xrightarrow{f} QY$ in $\text{Sets}$

$\rho \in D_m$

$\left( f(x)(y) \right)_{m,n}(\rho) \in D_n$

for each $n$
Quantum
Geometry of Interaction

$\left[ M \right] = M$

Hasuo (Tokyo)

Monday, November 7, 2011
Quantum

Geometry of Interaction

Not just a token/particle, but quantum state!

\[
[M] = M
\]

... (countably many)

Hasuo (Tokyo)

Monday, November 7, 2011
Quantum Geometry of Interaction

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

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

Not just a token/particle, but quantum state!

"Quantum Data"
Quantum Geometry of Interaction

\[ [M] = M \]

“Quantum Data”

Not just a token/particle, but quantum state!

“Classical Control”

(countably many)
Quantum Geometry of Interaction

\[[M] = M\]

“Quantum Data”

“Classical Control”

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

Not just a token/particle, but quantum state!
Indeed...

- The monad $Q$ qualifies as a "branching monad"
- The quantum GoI workflow leads to a linear category $\text{PER}_Q$
- From which we construct an adequate denotational model
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 :-)
Conclusion: 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

Quantum branching monad

Quantum TSMC

Quantum LCA

Model of quantum language

Monday, November 7, 2011
Conclusion: the Categorical GoI Workflow

- Linear combinatory algebra
- Realizability
- Coalgebraic trace semantics
- Traced monoidal category \( \mathbb{C} \)
- Categorical GoI [AHS02]
- Linear category
- Branching monad \( B \)

---

Thank you for your attention!
Ichiro Hasuo (Dept. CS, U Tokyo)
http://www-mmm.is.s.u-tokyo.ac.jp/~ichiro/

Quantum branching monad
Quantum TSMC
Quantum LCA
Model of quantum language

Monday, November 7, 2011
The Language $q\lambda^e$

- Roughly: linear $\lambda$ + quantum primitives
- “Quantum data, classical control”
- No superposed threads
- Based on [Selinger&Valiron’09]
- With slight modifications
- Notably: quantum $\otimes$ vs. linear logic $\otimes$
- The same in [Selinger&Valiron’09]
  - clean type system, aids programming
- But... problem with GoI-style semantics
The Language $q\lambda_e$

The *types* of $q\lambda_e$ are:

\[
A, B ::= n\text{-}qbit \mid ! A \mid A \rightarrow B \mid \top \mid A \otimes B \mid A + B,\]

with conventions $q\text{bit} := 1\text{-}qbit$ and $\text{bit} := \top + \top$.

The *terms* of $q\lambda_e$ are:

\[
M, N, P ::= \\
 x \mid \lambda x^A \cdot M \mid MN \mid \langle M, N \rangle \mid * \mid \\
\text{let } \langle x^A, y^B \rangle = M \text{ in } N \mid \text{let } * = M \text{ in } N \mid \\
\text{inj}_B \ M \mid \text{inj}_A \ M \mid \\
\text{match } P \text{ with } (x^A \leftrightarrow M \mid y^B \leftrightarrow N) \mid \\
\text{letrec } f^A x = M \text{ in } N \mid \\
\text{new } |0\rangle \mid \text{meas}_{i}^{n+1} \mid U \mid \text{cmp}_{m,n},
\]

with conventions $\text{tt} := \text{inj}_\ell^\top(*)$ and $\text{ff} := \text{inj}_r^\top(*)$. 
The **types** of $q\lambda_e$ are:

$$A, B ::= n\text{-qbit} \mid !A \mid A \rightarrow B \mid \top \mid A \otimes B \mid A + B,$$

with conventions $\text{qbit} := 1\text{-qbit}$ and $\text{bit} := \top + \top$.

The **terms** of $q\lambda_e$ are:

$$M, N, P ::= $x \mid \lambda x^A.M \mid MN \mid \langle M, N \rangle \mid * \mid$$

let $\langle x^A, y^B \rangle = M$ in $N$ \mid let $\ast = M$ in $N$ \mid

$\text{inj}^B_M \mid \text{inj}^A_M \mid$

match $P$ with $(x^A \mapsto M \mid y^B \mapsto N) \mid$

letrec $f^A x = M$ in $N$ \mid

new $|0\rangle \mid \text{meas}^{n+1}_i \mid U \mid \text{cmp}_{m,n}$,

with conventions $\text{tt} := \text{inj}^\top_\ell(*)$ and $\text{ff} := \text{inj}^\top_r(*)$.
2-qbit \cong \text{qbit} \otimes \text{qbit}

\[ A, B ::= n\text{-qbit} \mid !A \mid A \to B \mid \top \mid A \boxtimes B \mid A + B , \]

with conventions \( \text{qbit} := 1\text{-qbit} \) and \( \text{bit} := \top + \top \).

The terms of \( \text{q}\lambda\ell \) are:

\[ M, N, P ::= \]

\[ x \mid \lambda x^A.M \mid MN \mid \langle M, N \rangle \mid * \mid \]

\[ \text{let} \langle x^A, y^B \rangle = M \text{ in } N \mid \text{let } * = M \text{ in } N \mid \]

\[ \text{inj}^B_M \mid \text{inj}^A_M \mid \]

\[ \text{match } P \text{ with } (x^A \mapsto M \mid y^B \mapsto N) \mid \]

\[ \text{letrec } f^A x = M \text{ in } N \mid \]

\[ \text{new } |0\rangle \mid \text{meas}^{n+1}_i \mid U \mid \text{cmp}_{m,n} , \]

with conventions \( \text{tt} := \text{inj}_\ell^\top(*) \) and \( \text{ff} := \text{inj}_r^\top(*) \).
The Language

\[ A, B ::= n\text{-qbit} \mid !A \mid A \to B \mid \top \mid A \otimes B \mid A + B , \]
with conventions \( \text{qbit} := 1\text{-qbit} \) and \( \text{bit} := \top + \top \).

The terms of \( \lambda_\ell \) are:

\[ M, N, P ::= \]
\[ x \mid \lambda x^A.M \mid MN \mid \langle M, N \rangle \mid * \mid \]
let \( \langle x^A, y^B \rangle = M \) in \( N \) | let \( * = M \) in \( N \) |
\[ \text{inj}_B^A \ M \mid \text{inj}_r^A \ M \mid \]
match \( P \) with \( (x^A \mapsto M \mid y^B \mapsto N) \) |
letrec \( f^A x = M \) in \( N \) |
\[ \text{new} |0\rangle \mid \text{meas}_{i}^{n+1} |U \mid \text{cmp}_{m,n} , \]
with conventions \( \text{tt} := \text{inj}_\ell^\top(*) \) and \( \text{ff} := \text{inj}_r^\top(*) \).

Different from quantum \( \otimes \) (Unlike [Selinger-Valiron’09]); same as the one in PER.

Recursion

2-qbit \( \cong \text{qbit} \otimes \text{qbit} \)
The Language

2-qbit $\cong$ qbit $\otimes$ qbit

$$A, B ::= n\text{-qbit} \mid !A \mid A \rightarrow B \mid \top \mid A \otimes B \mid A + B,$$

with conventions qbit $:= 1$-qbit and bit $:= \top + \top$.

The terms of $q\lambda_\ell$ are:

$$M, N, P ::= $$

- $x$ | $\lambda x^A. M$ | $MN$ | $\langle M, N \rangle$ | $*$ |
- let $\langle x^A, y^B \rangle = M$ in $N$ | let $* = M$ in $N$ |
- inj$_B^A$ $M$ | inj$_r^A$ $M$ |
- match $P$ with $(x^A \mapsto M \mid y^B \mapsto N)$ |
- letrec $f^A x = M$ in $N$ |
- new $|0\rangle$ | meas$_i^{n+1}$ $|U\rangle$ | cmp$_{m,n}$ |

with conventions $\text{tt} := \text{inj}_\ell^\top(*)$ and $\text{ff} := \text{inj}_r^\top(*)$. 

Recursion

Quantum primitives

Different from quantum $\otimes$ (Unlike [Selinger-Valiron’09]); same as the one in PER

Monday, November 7, 2011
Implicit linearity tracking via subtyping <:

e.g. !A <: A, !A <: !!A

(following [Selinger-Valiron’09])

\[ n = 0 \Rightarrow m = 0 \] (T)
\[ \mathsf{\text{k-qbit}} <: !^m \mathsf{\text{k-qbit}} \] (k-qbit)
\[ A_1 <: B_1 \quad A_2 <: B_2 \] (k-qbit)
\[ !^m(A_1 \otimes A_2) <: !^m(B_1 \otimes B_2) \] (\#) with \# \in \{\otimes, +\}

Measurements

- \( A_{\text{new}0} := \text{qbit} \)
- \( A_{\text{meas}i} := (n+1)\text{-qbit} \rightarrow (\text{bit} \otimes n\text{-qbit}) \) for \( n \geq 1 \)
- \( A_{\text{meas}1} := \text{qbit} \rightarrow \text{bit} \)
- \( A_U := n\text{-qbit} \rightarrow n\text{-qbit} \) for a \( 2^n \times 2^n \) matrix \( U \)
- \( A_{\text{cmp}_{m,n}} := (m\text{-qbit} \otimes n\text{-qbit}) \rightarrow (m+n)\text{-qbit} \)

Bookkeeping (due to \( \otimes \) vs. \( \otimes \))

Hasuo (Tokyo)
#### Operational Semantics

- $E[(\lambda x^A.M)V] \rightarrow_1 E[M[V/x]]$
- $E[\text{let } \langle x^A, y^B \rangle = \langle V, W \rangle \text{ in } M] \rightarrow_1 E[M[V/x, W/y]]$
- $E[\text{let } * = * \text{ in } M] \rightarrow_1 E[M]$
- $E[\text{match } (\text{inj}_l^B V) \text{ with } (x^m_A \mapsto M \mid y^m_B \mapsto N)] \rightarrow_1 E[M[V/x]]$
- $E[\text{match } (\text{inj}_r^A V) \text{ with } (x^m_A \mapsto M \mid y^m_B \mapsto N)] \rightarrow_1 E[N[V/y]]$
- $E[\text{letrec } f^{A\rightarrow B} x = M \text{ in } N] \rightarrow_1 E[N[\lambda x^A.\text{letrec } f^{A\rightarrow B} x = M \text{ in } M/f]]$
- $E[\text{meas}_i^{n+1}(\text{new } \rho)] \rightarrow_1 E[\langle \text{tt}, \text{new } \langle 0_i|\rho|0_i \rangle \rangle]]$
- $E[\text{meas}_i^{n+1}(\text{new } \rho)] \rightarrow_1 E[\langle \text{ff}, \text{new } \langle 1_i|\rho|1_i \rangle \rangle]$