Categorical Semantics of Reversible Pattern-Matching

Louis Lemonnier, Benoît Valiron, Kostia Chardonnet

QuaCS

LMF

Laboratoire
Méthodes
Formelles

November 4th, 2021

SCALP 2021

Reversible programming

Reversibility

- Reversible computation: only apply bijections.

[1] Rolf Landauer (1961): Irreversibility and Heat Generation in the Computing Process. IBM Journal of Research and Development. 5(3), pp. 183–191, doi:10.1147/rd.53.0183.

Reversibility

- Reversible computation: only apply bijections.
- Application in low-consumption hardware [1].

[1] Rolf Landauer (1961): Irreversibility and Heat Generation in the Computing Process. IBM Journal of Research and Development. 5(3), pp. 183–191, doi:10.1147/rd.53.0183.

Reversibility

- Reversible computation: only apply bijections.
- Application in low-consumption hardware [1].
- Application in quantum computing, where operations (except measurement) are reversible.

[1] Rolf Landauer (1961): Irreversibility and Heat Generation in the Computing Process. IBM Journal of Research and Development. 5(3), pp. 183–191, doi:10.1147/rd.53.0183.

Reversibility

- Reversible computation: only apply bijections.
- Application in low-consumption hardware [1].
- Application in quantum computing, where operations (except measurement) are reversible.

What we are focusing on today:

[1] Rolf Landauer (1961): Irreversibility and Heat Generation in the Computing Process. IBM Journal of Research and Development. 5(3), pp. 183–191, doi:10.1147/rd.53.0183.

Reversibility

- Reversible computation: only apply bijections.
- Application in low-consumption hardware [1].
- Application in quantum computing, where operations (except measurement) are reversible.

What we are focusing on today:

- Reversible programming language

[1] Rolf Landauer (1961): Irreversibility and Heat Generation in the Computing Process. IBM Journal of Research and Development. 5(3), pp. 183–191, doi:10.1147/rd.53.0183.

Reversibility

- Reversible computation: only apply bijections.
- Application in low-consumption hardware [1].
- Application in quantum computing, where operations (except measurement) are reversible.

What we are focusing on today:

- Reversible programming language
- Detailed denotational semantics

[1] Rolf Landauer (1961): Irreversibility and Heat Generation in the Computing Process. IBM Journal of Research and Development. 5(3), pp. 183–191, doi:10.1147/rd.53.0183.

Terms and types

$$
\begin{array}{llcl}
\text{(Value types)} & a, b & ::= & \alpha \mid a \oplus b \mid a \otimes b \\
\text{(Iso types)} & T & ::= & a \leftrightarrow b \\
\text{(Values)} & v & ::= & c_\alpha \mid x \mid \mathtt{inj}_l\, v \mid \mathtt{inj}_r\, v \mid \langle v_1, v_2 \rangle \\
\text{(Functions)} & \omega & ::= & \{ \mid v_1 \leftrightarrow v'_1 \mid \ldots \mid v_n \leftrightarrow v'_n \} \\
\text{(Terms)} & t & ::= & v \mid \omega\, t
\end{array}
$$

Terms and types

$$
\begin{array}{llcl}
\text{(Value types)} & a, b & ::= & \alpha \mid a \oplus b \mid a \otimes b \\
\text{(Iso types)} & T & ::= & a \leftrightarrow b \\
\text{(Values)} & v & ::= & c_\alpha \mid x \mid \mathtt{inj}_l\, v \mid \mathtt{inj}_r\, v \mid \langle v_1, v_2 \rangle \\
\text{(Functions)} & \omega & ::= & \{ \mid v_1 \leftrightarrow v_1' \mid \dots \mid v_n \leftrightarrow v_n' \} \\
\text{(Terms)} & t & ::= & v \mid \omega\, t
\end{array}
$$

$$
\{ \mid \mathtt{inj}_l\, x \leftrightarrow \mathtt{inj}_r\, x \mid \mathtt{inj}_r\, x \leftrightarrow \mathtt{inj}_l\, x \}
$$

Terms and types

$$
\begin{array}{lrcl}
\text{(Value types)} & a, b & ::= & \alpha \mid a \oplus b \mid a \otimes b \\
\text{(Iso types)} & T & ::= & a \leftrightarrow b \\
\text{(Values)} & v & ::= & c_\alpha \mid x \mid \texttt{inj}_l\, v \mid \texttt{inj}_r\, v \mid \langle v_1, v_2 \rangle \\
\text{(Functions)} & \omega & ::= & \{ \mid v_1 \leftrightarrow v_1' \mid \ldots \mid v_n \leftrightarrow v_n' \} \\
\text{(Terms)} & t & ::= & v \mid \omega\, t
\end{array}
$$

$$
\{ \mid \texttt{inj}_l\, x \leftrightarrow \texttt{inj}_r\, x \mid \texttt{inj}_r\, x \leftrightarrow \texttt{inj}_l\, x \}
$$

- Two partial morphisms *joined*

Terms and types

$$
\begin{array}{lrcl}
\text{(Value types)} & a, b & ::= & \alpha \mid a \oplus b \mid a \otimes b \\
\text{(Iso types)} & T & ::= & a \leftrightarrow b \\
\text{(Values)} & v & ::= & c_\alpha \mid x \mid \texttt{inj}_l\, v \mid \texttt{inj}_r\, v \mid \langle v_1, v_2 \rangle \\
\text{(Functions)} & \omega & ::= & \{ \mid v_1 \leftrightarrow v_1' \mid \ldots \mid v_n \leftrightarrow v_n' \} \\
\text{(Terms)} & t & ::= & v \mid \omega\, t
\end{array}
$$

$$
\{ \mid \texttt{inj}_l\, x \leftrightarrow \texttt{inj}_r\, x \mid \texttt{inj}_r\, x \leftrightarrow \texttt{inj}_l\, x \}
$$

- Two partial morphisms *joined*
- With compatible domains

Terms and types

$$
\begin{array}{lrcl}
\text{(Value types)} & a, b & ::= & \alpha \mid a \oplus b \mid a \otimes b \\
\text{(Iso types)} & T & ::= & a \leftrightarrow b \\
\text{(Values)} & v & ::= & c_\alpha \mid x \mid \texttt{inj}_l\, v \mid \texttt{inj}_r\, v \mid \langle v_1, v_2 \rangle \\
\text{(Functions)} & \omega & ::= & \{ \mid v_1 \leftrightarrow v_1' \mid \ldots \mid v_n \leftrightarrow v_n' \} \\
\text{(Terms)} & t & ::= & v \mid \omega\, t
\end{array}
$$

$$
\{ \mid \texttt{inj}_l\, x \leftrightarrow \texttt{inj}_r\, x \mid \texttt{inj}_r\, x \leftrightarrow \texttt{inj}_l\, x \}
$$

- Two partial morphisms *joined*
- With compatible domains
- Which inverses have compatible domains

Terms and types

$$
\begin{array}{llcl}
\text{(Value types)} & a, b & ::= & \alpha \mid a \oplus b \mid a \otimes b \\
\text{(Iso types)} & T & ::= & a \leftrightarrow b \\
\text{(Values)} & v & ::= & c_\alpha \mid x \mid \texttt{inj}_l\, v \mid \texttt{inj}_r\, v \mid \langle v_1, v_2 \rangle \\
\text{(Functions)} & \omega & ::= & \{ \mid v_1 \leftrightarrow v_1' \mid \ldots \mid v_n \leftrightarrow v_n' \} \\
\text{(Terms)} & t & ::= & v \mid \omega\, t
\end{array}
$$

$$\{ \mid \texttt{inj}_l\, x \leftrightarrow \texttt{inj}_r\, x \mid \texttt{inj}_r\, x \leftrightarrow \texttt{inj}_l\, x \}$$

- Two partial morphisms *joined*
- With compatible domains
- Which inverses have compatible domains

Operational semantics:

$$\{ \mid \texttt{inj}_l\, x \leftrightarrow \texttt{inj}_r\, x \mid \texttt{inj}_r\, x \leftrightarrow \texttt{inj}_l\, x \} (\texttt{inj}_r\, \star) \longrightarrow \texttt{inj}_l\, \star$$

Categorical model

Restriction and inverse category

Domain and partiality

Partial inverse

Compatibility

Union of partial isos

Restriction and inverse category

Domain and partiality   Restriction category: $f \mapsto \bar{f}$.

Partial inverse

Compatibility

Union of partial isos

Restriction and inverse category

Domain and partiality

Restriction category: $f \mapsto \bar{f}$.

$f \circ \bar{f} = f$, $\bar{f} \circ \bar{g} = \bar{g} \circ \bar{f}$, $\overline{f \circ \bar{g}} = \bar{f} \circ \bar{g}$, $\bar{h} \circ f = f \circ \overline{h \circ f}$.

Partial inverse

Compatibility

Union of partial isos

Restriction and inverse category

Domain and partiality

Restriction category: $f \mapsto \bar{f}$.

$f \circ \bar{f} = f, \quad \bar{f} \circ \bar{g} = \bar{g} \circ \bar{f}, \quad \overline{f \circ \bar{g}} = \bar{f} \circ \bar{g}, \quad \bar{h} \circ f = f \circ \overline{h \circ f}.$

Partial inverse

Compatibility

Union of partial isos

Restriction and inverse category

Domain and partiality — Restriction category: $f \mapsto \bar{f}$.

$f \circ \bar{f} = f$, $\bar{f} \circ \bar{g} = \bar{g} \circ \bar{f}$, $\overline{f \circ \bar{g}} = \bar{f} \circ \bar{g}$, $\bar{h} \circ f = f \circ \overline{h \circ f}$.

Partial inverse — Inverse category: $f \mapsto f^{\circ}$. $f^{\circ} \circ f = \bar{f}$ and $f \circ f^{\circ} = \overline{f^{\circ}}$.

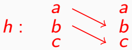
Compatibility

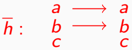
Union of partial isos

Restriction and inverse category

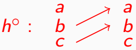
Domain and partiality — Restriction category: $f \mapsto \bar{f}$.

$f \circ \bar{f} = f$, $\bar{f} \circ \bar{g} = \bar{g} \circ \bar{f}$, $\overline{f \circ \bar{g}} = \bar{f} \circ \bar{g}$, $\bar{h} \circ f = f \circ \overline{h \circ f}$.





Partial inverse — Inverse category: $f \mapsto f^{\circ}$. $f^{\circ} \circ f = \bar{f}$ and $f \circ f^{\circ} = \overline{f^{\circ}}$.



Compatibility

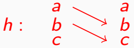
Union of partial isos

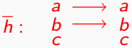
Restriction and inverse category

Domain and partiality

Restriction category: $f \mapsto \bar{f}$.

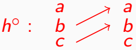
$f \circ \bar{f} = f$, $\bar{f} \circ \bar{g} = \bar{g} \circ \bar{f}$, $\overline{f \circ \bar{g}} = \bar{f} \circ \bar{g}$, $\bar{h} \circ f = f \circ \overline{h \circ f}$.





Partial inverse

Inverse category: $f \mapsto f^{\circ}$. $f^{\circ} \circ f = \bar{f}$ and $f \circ f^{\circ} = \overline{f^{\circ}}$.



Compatibility

Restriction compatible:

$f \smile g : f\bar{g} = g\bar{f}$, $f \asymp g : f \smile g$ and $f^{\circ} \smile g^{\circ}$.

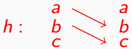
Union of partial isos

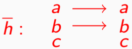
Restriction and inverse category

Domain and partiality

Restriction category: $f \mapsto \bar{f}$.

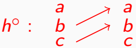
$f \circ \bar{f} = f$, $\bar{f} \circ \bar{g} = \bar{g} \circ \bar{f}$, $\overline{f \circ \bar{g}} = \bar{f} \circ \bar{g}$, $\bar{h} \circ f = f \circ \overline{h \circ f}$.





Partial inverse

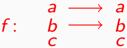
Inverse category: $f \mapsto f^{\circ}$. $f^{\circ} \circ f = \bar{f}$ and $f \circ f^{\circ} = \overline{f^{\circ}}$.

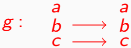


Compatibility

Restriction compatible:

$f \smile g : f\bar{g} = g\bar{f}$, $f \asymp g : f \smile g$ and $f^{\circ} \smile g^{\circ}$.



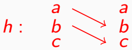


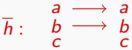
Union of partial isos

Restriction and inverse category

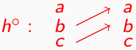
Domain and partiality — Restriction category: $f \mapsto \bar{f}$.

$f \circ \bar{f} = f$, $\bar{f} \circ \bar{g} = \bar{g} \circ \bar{f}$, $\overline{f \circ \bar{g}} = \bar{f} \circ \bar{g}$, $\bar{h} \circ f = f \circ \overline{h \circ f}$.



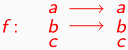


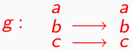
Partial inverse — Inverse category: $f \mapsto f^\circ$. $f^\circ \circ f = \bar{f}$ and $f \circ f^\circ = \overline{f^\circ}$.



Compatibility — Restriction compatible:

$f \smile g : f\bar{g} = g\bar{f}$, $f \asymp g : f \smile g$ and $f^\circ \smile g^\circ$.





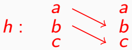
Union of partial isos — Partial order: $f \leq g : g\bar{f} = f$.

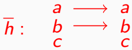
Restriction and inverse category

Domain and partiality

Restriction category: $f \mapsto \bar{f}$.

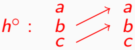
$f \circ \bar{f} = f$, $\bar{f} \circ \bar{g} = \bar{g} \circ \bar{f}$, $\overline{f \circ \bar{g}} = \bar{f} \circ \bar{g}$, $\bar{h} \circ f = f \circ \overline{h \circ f}$.





Partial inverse

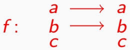
Inverse category: $f \mapsto f^\circ$. $f^\circ \circ f = \bar{f}$ and $f \circ f^\circ = \overline{f^\circ}$.

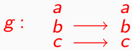


Compatibility

Restriction compatible:

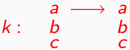
$f \smile g : f\bar{g} = g\bar{f}$, $f \asymp g : f \smile g$ and $f^\circ \smile g^\circ$.





Union of partial isos

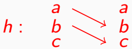
Partial order: $f \leq g : g\bar{f} = f$.

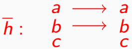


Restriction and inverse category

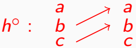
Domain and partiality — Restriction category: $f \mapsto \bar{f}$.

$f \circ \bar{f} = f$, $\bar{f} \circ \bar{g} = \bar{g} \circ \bar{f}$, $\overline{f \circ \bar{g}} = \bar{f} \circ \bar{g}$, $\bar{h} \circ f = f \circ \overline{h \circ f}$.



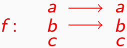


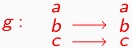
Partial inverse — Inverse category: $f \mapsto f^{\circ}$. $f^{\circ} \circ f = \bar{f}$ and $f \circ f^{\circ} = \overline{f^{\circ}}$.



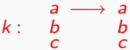
Compatibility — Restriction compatible:

$f \smile g : f\bar{g} = g\bar{f}$, $f \asymp g : f \smile g$ and $f^{\circ} \smile g^{\circ}$.





Union of partial isos — Partial order: $f \leq g : g\bar{f} = f$.



Join: $\bigvee_{s \in S} s$.

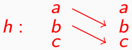
if $s \leq t$, then $s \leq \bigvee_{s \in S} s$, $\bigvee_{s \in S} s \leq t$, $\overline{\bigvee_{s \in S} s} = \bigvee_{s \in S} \bar{s}$,

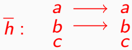
Restriction and inverse category

Domain and partiality

Restriction category: $f \mapsto \bar{f}$.

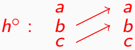
$f \circ \bar{f} = f$, $\bar{f} \circ \bar{g} = \bar{g} \circ \bar{f}$, $\overline{f \circ \bar{g}} = \bar{f} \circ \bar{g}$, $\bar{h} \circ f = f \circ \overline{h \circ f}$.





Partial inverse

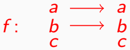
Inverse category: $f \mapsto f^{\circ}$. $f^{\circ} \circ f = \bar{f}$ and $f \circ f^{\circ} = \overline{f^{\circ}}$.

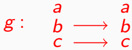


Compatibility

Restriction compatible:

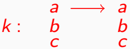
$f \smile g : f\bar{g} = g\bar{f}$, $f \asymp g : f \smile g$ and $f^{\circ} \smile g^{\circ}$.





Union of partial isos

Partial order: $f \leq g : g\bar{f} = f$.



Join: $\bigvee_{s \in S} s$.

if $s \leq t$, then $s \leq \bigvee_{s \in S} s$, $\bigvee_{s \in S} s \leq t$, $\overline{\bigvee_{s \in S} s} = \bigvee_{s \in S} \bar{s}$,

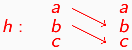
$$f \circ \left(\bigvee_{s \in S} s \right) = \bigvee_{s \in S} fs, \left(\bigvee_{s \in S} s \right) \circ g = \bigvee_{s \in S} sg.$$

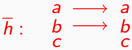
Restriction and inverse category

Domain and partiality

Restriction category: $f \mapsto \bar{f}$.

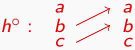
$f \circ \bar{f} = f$, $\bar{f} \circ \bar{g} = \bar{g} \circ \bar{f}$, $\overline{f \circ \bar{g}} = \bar{f} \circ \bar{g}$, $\bar{h} \circ f = f \circ \overline{h \circ f}$.





Partial inverse

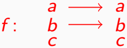
Inverse category: $f \mapsto f^\circ$. $f^\circ \circ f = \bar{f}$ and $f \circ f^\circ = \overline{f^\circ}$.

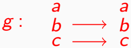


Compatibility

Restriction compatible:

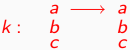
$f \smile g : f\bar{g} = g\bar{f}$, $f \asymp g : f \smile g$ and $f^\circ \smile g^\circ$.





Union of partial isos

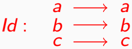
Partial order: $f \leq g : g\bar{f} = f$.



Join: $\bigvee_{s \in S} s$.

if $s \leq t$, then $s \leq \bigvee_{s \in S} s$, $\bigvee_{s \in S} s \leq t$, $\overline{\bigvee_{s \in S} s} = \bigvee_{s \in S} \bar{s}$,

$$f \circ \left(\bigvee_{s \in S} s \right) = \bigvee_{s \in S} fs, \left(\bigvee_{s \in S} s \right) \circ g = \bigvee_{s \in S} sg.$$



PId

Example

S a set. PId_S category with one object $*$ and morphisms $Y \subseteq S$.

- Composition: intersection
- Identity: S
- Restriction of Y: Y
- Partial inverse of Y: Y
- Join: union.

Denotational semantics

Terms and values

$[\![\Delta]\!] = [\![a_1]\!] \otimes \cdots \otimes [\![a_n]\!]$ whenever $\Delta \doteq x_1 : a_1, \ldots, x_n : a_n$.

$[\![\Delta \vdash \texttt{inj}_l\, v : a \oplus b]\!] \doteq \iota_l \circ f$, whenever $f = [\![\Delta \vdash v : a]\!]$, and similarly for the right-projection.

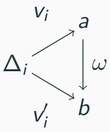
If $f = [\![\Delta_1 \vdash v_1 : a_1]\!]$ and $g = [\![\Delta_2 \vdash v_2 : a_2]\!]$,
$[\![\Delta_1, \Delta_2 \vdash \langle v_1, v_2 \rangle : a_1 \otimes a_2]\!] \doteq f \otimes g$.

Isos

An iso: $\{ \; | \; v_1 \leftrightarrow v_1' \mid \ldots \mid \; v_n \leftrightarrow v_n' \; \} : a \leftrightarrow b$.

An iso: $\{ \ \mid \ v_1 \leftrightarrow v'_1 \mid \dots \mid \ v_n \leftrightarrow v'_n \ \} : a \leftrightarrow b$.

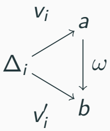
The situation:



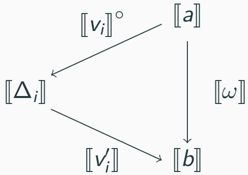
Isos

An iso: $\{ \mid v_1 \leftrightarrow v'_1 \mid \ldots \mid v_n \leftrightarrow v'_n \} : a \leftrightarrow b$.

The situation:



What we'd like to build:



Careful: this diagram does not commute!

Isos

Lemma

If $v_1 \perp v_2$ *and* $v'_1 \perp v'_2$, *then* $[\![v'_1]\!] \circ [\![v_1]\!]^\circ \asymp [\![v'_2]\!] \circ [\![v_2]\!]^\circ$.

Isos

Lemma

If $v_1 \perp v_2$ *and* $v_1' \perp v_2'$*, then* $[\![v_1']\!] \circ [\![v_1]\!]^\circ \asymp [\![v_2']\!] \circ [\![v_2]\!]^\circ$.

The definition of isos in the language involves these orthogonalities, thus all $[\![v_i']\!] \circ [\![v_i]\!]^\circ$ form a compatible set.

Isos

Lemma

If $v_1 \perp v_2$ *and* $v'_1 \perp v'_2$, *then* $[\![v'_1]\!] \circ [\![v_1]\!]^\circ \asymp [\![v'_2]\!] \circ [\![v_2]\!]^\circ$.

The definition of isos in the language involves these orthogonalities, thus all $[\![v'_i]\!] \circ [\![v_i]\!]^\circ$ form a compatible set.

Definition

$$[\![\vdash_\omega \{ \ \mid \ v_1 \leftrightarrow v'_1 \mid \ v_2 \leftrightarrow v'_2 \ \ldots \ \} : a \leftrightarrow b]\!] = \bigvee_i [\![v'_i]\!] \circ [\![v_i]\!]^\circ : [\![a]\!] \to [\![b]\!] \,.$$

What we would like to have thus

The language is based is pattern-matching:

$$\vdash_\omega \{ \ \mid \ v_1 \leftrightarrow v_1' \mid \ v_2 \leftrightarrow v_2' \ \dots \ \}$$

decidable which pattern a term u fits.
It should also be in the denotational semantics:

$$\big((\llbracket v_1' \rrbracket \circ \llbracket v_1 \rrbracket^\circ) \vee (\llbracket v_2' \rrbracket \circ \llbracket v_2 \rrbracket^\circ)\big) \circ \llbracket u \rrbracket = \llbracket v_i' \rrbracket \circ \llbracket v_i \rrbracket^\circ \circ \llbracket u \rrbracket$$

for some i.

What we would like to have thus

The language is based is pattern-matching:

$$\vdash_\omega \{ \ | \ v_1 \leftrightarrow v_1' \mid \ v_2 \leftrightarrow v_2' \ \dots \ \}$$

decidable which pattern a term u fits.
It should also be in the denotational semantics:

$$\big((\llbracket v_1' \rrbracket \circ \llbracket v_1 \rrbracket^\circ) \vee (\llbracket v_2' \rrbracket \circ \llbracket v_2 \rrbracket^\circ)\big) \circ \llbracket u \rrbracket = \llbracket v_i' \rrbracket \circ \llbracket v_i \rrbracket^\circ \circ \llbracket u \rrbracket$$

for some i.
Usually, it relies on disjointness ($\oplus$).

Pattern-matching and consistency

The origin of consistency

In $(f \vee g)h$, we want h to choose:

$$fh \vee gh = fh \text{ or } fh \vee gh = gh$$

A join inverse category with this property will be called *pattern-matching* category.

The origin of consistency

In $(f \vee g)h$, we want h to choose:

$$fh \vee gh = fh \text{ or } fh \vee gh = gh$$

A join inverse category with this property will be called *pattern-matching* category. With disjointness ($\oplus$), rather $fh = 0$ or $gh = 0$.

The origin of consistency

In $(f \vee g)h$, we want h to choose:

$$fh \vee gh = fh \text{ or } fh \vee gh = gh$$

A join inverse category with this property will be called *pattern-matching* category. With disjointness ($\oplus$), rather $fh = 0$ or $gh = 0$.

The idea: translate it as a notion of non decomposability, to have $fh \vee gh$ not decomposable.

For this, we need h to be not decomposable either.

The origin of consistency

In $(f \vee g)h$, we want h to choose:

$$fh \vee gh = fh \text{ or } fh \vee gh = gh$$

A join inverse category with this property will be called *pattern-matching* category. With disjointness ($\oplus$), rather $fh = 0$ or $gh = 0$.

The idea: translate it as a notion of non decomposability, to have $fh \vee gh$ not decomposable.

For this, we need h to be not decomposable either.

Consistency: for any morphism k, if h is non decomposable, so is kh.

A join inverse category with this property will be called *consistent* category.

The origin of consistency

In $(f \vee g)h$, we want h to choose:

$$fh \vee gh = fh \text{ or } fh \vee gh = gh$$

A join inverse category with this property will be called *pattern-matching* category. With disjointness ($\oplus$), rather $fh = 0$ or $gh = 0$.
The idea: translate it as a notion of non decomposability, to have $fh \vee gh$ not decomposable.
For this, we need h to be not decomposable either.
Consistency: for any morphism k, if h is non decomposable, so is kh.
A join inverse category with this property will be called *consistent* category.

Consistency implies pattern-matching.

The origin of consistency

In $(f \vee g)h$, we want h to choose:

$$fh \vee gh = fh \text{ or } fh \vee gh = gh$$

A join inverse category with this property will be called *pattern-matching* category. With disjointness ($\oplus$), rather $fh = 0$ or $gh = 0$.

The idea: translate it as a notion of non decomposability, to have $fh \vee gh$ not decomposable.

For this, we need h to be not decomposable either.

Consistency: for any morphism k, if h is non decomposable, so is kh.

A join inverse category with this property will be called *consistent* category.

Consistency implies pattern-matching.

What about the other way around?

The different kinds

Definition (Strongly non decomposable)

$$0 \text{ —— } m$$

Thus $m = u \vee v \Rightarrow u = 0$ or $v = 0$

The different kinds

Definition (Strongly non decomposable)

$$0 \text{ --- } m$$

Thus $m = u \vee v \Rightarrow u = 0$ or $v = 0$

Definition (Linearly non decomposable)

$$0 \text{ ---} \cdots \text{--- } f \text{ --- } m$$

Thus $m = u \vee v \Rightarrow u \leq v$ or $v \leq u$

The different kinds

Definition (Strongly non decomposable)

$$0 \text{ --- } m$$

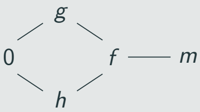
Thus $m = u \vee v \Rightarrow u = 0$ or $v = 0$

Definition (Linearly non decomposable)

$$0 \text{ ---} \cdots \text{--- } f \text{ --- } m$$

Thus $m = u \vee v \Rightarrow u \leq v$ or $v \leq u$

Definition (Weakly non decomposable)



Thus $m = u \vee v \Rightarrow u = m$ or $v = m$

The different kinds

Definition (Strongly non decomposable)

$$0 \text{ --- } m$$

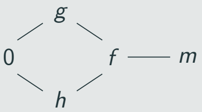
Thus $m = u \vee v \Rightarrow u = 0$ or $v = 0$

Definition (Linearly non decomposable)

$$0 \text{ ---} \cdots \text{--- } f \text{ --- } m$$

Thus $m = u \vee v \Rightarrow u \leq v$ or $v \leq u$

Definition (Weakly non decomposable)



Thus $m = u \vee v \Rightarrow u = m$ or $v = m$

strongly $\Rightarrow$ linearly $\Rightarrow$ weakly

The results

Theorem

A weakly pattern-matching category is weakly consistent.

The results

Theorem

A weakly pattern-matching category is weakly consistent.

Theorem

A join inverse category is strongly consistent.

The results

Theorem

A weakly pattern-matching category is weakly consistent.

Theorem

A join inverse category is strongly consistent.

Theorem

A join inverse category is linearly consistent.

The results

Theorem

A weakly pattern-matching category is weakly consistent.

Theorem

A join inverse category is strongly consistent.

Theorem

A join inverse category is linearly consistent.

This allows to not rely on a disjointness ($\oplus$) structure.

And to use a category like PId_S as a sound and adequate denotation.

The results

Theorem

A weakly pattern-matching category is weakly consistent.

Theorem

A join inverse category is strongly consistent.

Theorem

A join inverse category is linearly consistent.

This allows to not rely on a disjointness ($\oplus$) structure.

And to use a category like PId_S as a sound and adequate denotation.

The results

Theorem

A weakly pattern-matching category is weakly consistent.

Theorem

A join inverse category is strongly consistent.

Theorem

A join inverse category is linearly consistent.

This allows to not rely on a disjointness ($\oplus$) structure.

And to use a category like PId_S as a sound and adequate denotation.

Going further

Recursion

$\mathcal{C}$ a join inverse category.

For any A, B objects of $\mathcal{C}$, $\mathrm{Hom}_{\mathcal{C}}(A, B)$ is a DCPO.

Quantum programming language

Patterns are vectors in a basis of a Hilbert space.

Work in progress.

Thank you