Categorical Semantics of Reversible Pattern-Matching

Louis Lemonnier, Benoît Valiron, Kostia Chardonnet

November 4th, 2021

SCALP 2021
Reversible programming
Reversibility

- Reversible computation: only apply bijections.

---

Reversibility

- Reversible computation: only apply bijections.
- Application in low-consumption hardware \(^1\).

Reversibility

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

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:

---

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

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

Terms and types

(Value types) \( a, b ::= \alpha | a \oplus b | a \otimes b \)

(Iso types) \( T ::= a \leftrightarrow b \)

(Values) \( v ::= c_\alpha | x | \text{inj}_l v | \text{inj}_r v | \langle v_1, v_2 \rangle \)

(Functions) \( \omega ::= \{ | v_1 \leftrightarrow v'_1 | \ldots | v_n \leftrightarrow v'_n \} \)

(Terms) \( t ::= v | \omega t \)
Terms and types

(Value types) \( a, b \) ::= \( \alpha \mid a \oplus b \mid a \otimes b \)

(Iso types) \( T \) ::= \( a \leftrightarrow b \)

(Values) \( v \) ::= \( c_\alpha \mid x \mid \text{inj}_l v \mid \text{inj}_r v \mid \langle v_1, v_2 \rangle \)

(Functions) \( \omega \) ::= \{ \mid v_1 \leftrightarrow v'_1 \mid \ldots \mid v_n \leftrightarrow v'_n \}

(Terms) \( t \) ::= \( v \mid \omega t \)

\{ \mid \text{inj}_l x \leftrightarrow \text{inj}_r x \mid \text{inj}_r x \leftrightarrow \text{inj}_l x \}
Terms and types

(Value types) \[ a, b ::= \alpha \mid a \oplus b \mid a \otimes b \]

(Iso types) \[ T ::= a \leftrightarrow b \]

(Values) \[ v ::= c_{\alpha} \mid x \mid \text{inj}_l v \mid \text{inj}_r v \mid \langle v_1, v_2 \rangle \]

(Functions) \[ \omega ::= \{ \mid v_1 \leftrightarrow v'_1 \mid \ldots \mid v_n \leftrightarrow v'_n \} \]

(Terms) \[ t ::= v \mid \omega t \]

\[
\{ \mid \text{inj}_l x \leftrightarrow \text{inj}_r x \mid \text{inj}_r x \leftrightarrow \text{inj}_l x \}
\]

- Two partial morphisms joined
Terms and types

(Value types) \( a, b ::= \alpha | a \oplus b | a \otimes b \)

(Iso types) \( T ::= a \leftrightarrow b \)

(Values) \( v ::= c_{\alpha} | x | \text{inj}_l v | \text{inj}_r v | \langle v_1, v_2 \rangle \)

(Functions) \( \omega ::= \{ | v_1 \leftrightarrow v'_1 | \ldots | v_n \leftrightarrow v'_n \} \)

(Terms) \( t ::= v | \omega t \)

\[
\{ | \text{inj}_l x \leftrightarrow \text{inj}_r x | \text{inj}_r x \leftrightarrow \text{inj}_l x \}
\]

- Two partial morphisms \textit{joined}
- With compatible domains
Terms and types

(Value types) \( a, b ::= \alpha \mid a \oplus b \mid a \otimes b \)

(Iso types) \( T ::= a \leftrightarrow b \)

(Values) \( v ::= c_\alpha \mid x \mid \text{inj}_l v \mid \text{inj}_r v \mid \langle v_1, v_2 \rangle \)

(Functions) \( \omega ::= \{ \mid v_1 \leftrightarrow v'_1 \mid \ldots \mid v_n \leftrightarrow v'_n \} \)

(Terms) \( t ::= v \mid \omega t \)

\[
\{ \mid \text{inj}_l x \leftrightarrow \text{inj}_r x \mid \text{inj}_r x \leftrightarrow \text{inj}_l x \}
\]

- Two partial morphisms \textit{joined}
- With compatible domains
- Which inverses have compatible domains
Terms and types

(Value types) \( a, b ::= \alpha | a \oplus b | a \otimes b \)

(Iso types) \( T ::= a \leftrightarrow b \)

(Values) \( v ::= c_\alpha | x | \text{inj}_l v | \text{inj}_r v | \langle v_1, v_2 \rangle \)

(Functions) \( \omega ::= \{ | v_1 \leftrightarrow v_1' | \ldots | v_n \leftrightarrow v_n' \} \)

(Terms) \( t ::= v | \omega t \)

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

Operational semantics:
\[ \{ | \text{inj}_l x \leftrightarrow \text{inj}_r x | \text{inj}_r x \leftrightarrow \text{inj}_l x \} (\text{inj}_r \ast) \longrightarrow \text{inj}_l \ast \]
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: $f \mapsto \tilde{f}$.

Partial inverse

Compatibility

Union of partial isos
Restriction and inverse category

Domain and partiality

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

\[
\begin{align*}
  f \circ \bar{f} &= f, \\
  \bar{f} \circ \bar{g} &= \bar{g} \circ \bar{f}, \\
  \bar{f} \circ \bar{g} &= \bar{f} \circ \bar{g}, \\
  \bar{h} \circ f &= f \circ h \circ f.
\end{align*}
\]

Partial inverse

Compatibility

Union of partial isos
Restriction and inverse category

Domain and partiality

Restriction category: \( f \mapsto \overline{f} \).

\[
\begin{align*}
    f \circ \overline{f} &= f, \\
    \overline{f} \circ \overline{g} &= \overline{g} \circ \overline{f}, \\
    f \circ \overline{g} &= \overline{f} \circ \overline{g}, \\
    \overline{h} \circ f &= f \circ h \circ f.
\end{align*}
\]

Partial inverse

Compatibility

Union of partial isos
Restriction and inverse category

Domain and partiality

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

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

Partial inverse

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

Compatibility

Union of partial isos
Restriction and inverse category

Domain and partiality

Restriction category: \( f \mapsto \tilde{f} \).

\[ f \circ \tilde{f} = f, \quad \tilde{f} \circ g = g \circ \tilde{f}, \quad \tilde{f} \circ \tilde{g} = \tilde{f} \circ g, \quad \tilde{h} \circ f = f \circ \tilde{h} \circ f. \]

Partial inverse

Inverse category: \( f \mapsto f^\circ \).

\[ f^\circ \circ f = f, \quad f \circ f^\circ = f^\circ \circ f. \]

Compatibility

Union of partial isos
## Restriction and inverse category

### Domain and partiality

Restriction category: \( f \mapsto \overline{f} \).

\[
f \circ \overline{f} = f, \quad \overline{f} \circ g = \overline{g} \circ \overline{f}, \quad f \circ \overline{g} = \overline{f} \circ \overline{g}, \quad \overline{h} \circ f = f \circ \overline{h} \circ f.
\]

<table>
<thead>
<tr>
<th>Domain</th>
<th>( a )</th>
<th>( b )</th>
<th>( c )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( h ):</td>
<td>( b )</td>
<td>( c )</td>
<td>( a )</td>
</tr>
<tr>
<td>( \overline{h} ):</td>
<td>( b )</td>
<td>( c )</td>
<td>( a )</td>
</tr>
</tbody>
</table>

### Partial inverse

Inverse category: \( f \mapsto f^\circ \). \( f^\circ \circ f = \overline{f} \) and \( f \circ f^\circ = \overline{f} \).

<table>
<thead>
<tr>
<th>Domain</th>
<th>( a )</th>
<th>( b )</th>
<th>( c )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( h^\circ ):</td>
<td>( a )</td>
<td>( b )</td>
<td>( c )</td>
</tr>
</tbody>
</table>

### Compatibility

Restriction compatible:

\( f \bowtie g : f g = g f, \quad f \bowtie g : f \bowtie g \) and \( f^\circ \bowtie g^\circ \).

### Union of partial isos

Partial order:

\( f \leq g : g f = f \).

<table>
<thead>
<tr>
<th>Domain</th>
<th>( a )</th>
<th>( b )</th>
<th>( c )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( k ):</td>
<td>( a )</td>
<td>( b )</td>
<td>( c )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Domain</th>
<th>( a )</th>
<th>( b )</th>
<th>( c )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \lor ):</td>
<td>( a )</td>
<td>( b )</td>
<td>( c )</td>
</tr>
</tbody>
</table>

if \( s \leq t \), then \( s \leq \lor s \in S \lor t \leq t \), \( \lor s \in S \lor t \leq t \), \( \lor s \in S \lor t = \lor s \in S \lor t \), \( f \circ (\lor s \in S \lor t) = \lor s \in S f \), \( (\lor s \in S \lor t) \circ g = \lor s \in S sg \).
Restriction and inverse category

Domain and partiality

Restriction category: \( f \mapsto \bar{f} \).

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

Partial inverse

Inverse category: \( f \mapsto f^\circ \). \( f^\circ \circ f = \bar{f} \) and \( f \circ f^\circ = \bar{f}. \)

Compatibility

Restriction compatible:

\( f \bowtie g : f \bar{g} = g \bar{f}, \quad f \bowtie g : f \bowtie g \) and \( f^\circ \bowtie g^\circ \).

Union of partial isos
## Restriction and inverse category

### Domain and partiality

**Restriction category:** \( f \mapsto \tilde{f} \).

\[
\begin{align*}
f \circ \tilde{f} &= f, & f \circ \bar{g} &= \bar{g} \circ \bar{f}, & f \circ \bar{g} &= \tilde{f} \circ \bar{g}, & h \circ f &= f \circ h \circ f.
\end{align*}
\]

\[
\begin{array}{c}
h : \begin{array}{ccc}
a & \to & a \\
b & \to & b \\
c & \to & c \\
\end{array} & \quad \quad \quad \quad \quad \quad \quad
\bar{h} : \begin{array}{ccc}
a & \to & a \\
b & \to & b \\
c & \to & c \\
\end{array}
\end{array}
\]

### Partial inverse

**Inverse category:** \( f \mapsto f^\circ \). \( f^\circ \circ f = \tilde{f} \) and \( f \circ f^\circ = \bar{f} \).

\[
\begin{array}{c}
h^\circ : \begin{array}{ccc}
a & \to & a \\
b & \to & b \\
c & \to & c \\
\end{array}
\end{array}
\]

### Compatibility

**Restriction compatible:**

\[
f \bowtie g : f g = g \tilde{f}, \quad f \bowtie g : f \bowtie g \quad \text{and} \quad f^\circ \bowtie g^\circ.
\]

\[
\begin{array}{c}
f : \begin{array}{ccc}
a & \to & a \\
b & \to & b \\
c & \to & c \\
\end{array} & \quad \quad \quad \quad \quad \quad \quad
f : \begin{array}{ccc}
a & \to & a \\
b & \to & b \\
c & \to & c \\
\end{array} & \quad \quad \quad \quad \quad \quad \quad
f : \begin{array}{ccc}
a & \to & a \\
b & \to & b \\
c & \to & c \\
\end{array}
\end{array}
\]

### Union of partial isos

**Partial order:** \( f \leq g : g \tilde{f} = f \).
## Restriction and inverse category

### Domain and partiality

**Restriction category:** \( f \mapsto \bar{f}. \)

\[
f \circ \bar{f} = f, \quad \bar{f} \circ g = g \circ \bar{f}, \quad \bar{f} \circ g = \bar{f} \circ \bar{g}, \quad \bar{h} \circ f = f \circ \bar{h} \circ f.
\]

**Partial inverse**

**Inverse category:** \( f \mapsto f^\circ. \quad f^\circ \circ f = \bar{f} \) and \( f \circ f^\circ = \bar{f}. \)

### Partial inverse

**Compatibility**

**Restriction compatible:**

\[
f \leadsto g : f g = g f, \quad f \leadsto g : f \leadsto g \text{ and } f^\circ \leadsto g^\circ.
\]

**Union of partial isos**

**Partial order:** \( f \leq g : g \bar{f} = f. \)

### Union of partial isos

- \( f: a \rightarrow b, \quad b \rightarrow c, \quad g: a \rightarrow b, \quad b \rightarrow c \)
- \( k: b \rightarrow b, \quad c \rightarrow c \)
Restriction and inverse category

Domain and partiality

Restriction category: \( f \mapsto \tilde{f} \).

\[
\begin{align*}
    \tilde{f} \circ \tilde{f} &= \tilde{f}, & \tilde{f} \circ \tilde{g} &= \tilde{g} \circ \tilde{f}, & \tilde{f} \circ \tilde{g} &= \tilde{f} \circ \tilde{g}, & \tilde{h} \circ \tilde{f} &= \tilde{f} \circ \tilde{h} \circ \tilde{f}.
\end{align*}
\]

Partial inverse

Inverse category: \( f \mapsto f^\circ \). \( f^\circ \circ f = \tilde{f} \) and \( f \circ f^\circ = \tilde{f} \).

Compatibility

Restriction compatible:

\[
\begin{align*}
f \sim g : f \tilde{g} = g \tilde{f}, & \quad f \sim g : f \sim g \text{ and } f^\circ \sim g^\circ.
\end{align*}
\]

Union of partial isos

Partial order: \( f \leq g : g \tilde{f} = f \).

Join:

\[
\begin{align*}
    \bigvee_{s \in S} s.
\end{align*}
\]

if \( s \leq t \), then \( s \leq \bigvee_{s \in S} s, \bigvee_{s \in S} s \leq t, \bigvee_{s \in S} s = \bigvee_{s \in S} \tilde{s}, \)
## Restriction and inverse category

### Domain and partiality

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

\[
\begin{align*}
f \circ \bar{f} &= f, & f \circ \bar{g} &= \bar{g} \circ \bar{f}, & f \circ \bar{g} &= \bar{f} \circ \bar{g}, & h \circ f &= f \circ h \circ f. \\
h: \quad a & \rightarrow b & a & \rightarrow b & a & \rightarrow a \\
& \downarrow & & \downarrow & & \\
& b & \rightarrow c & b & \rightarrow c & b & \rightarrow b \\
\end{align*}
\]

### Partial inverse

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

\[
\begin{align*}
h^\circ: \quad a & \rightarrow b & a & \rightarrow a \\
& \downarrow & & \downarrow & & \\
& b & \rightarrow c & b & \rightarrow b \\
\end{align*}
\]

### Compatibility

**Restriction compatible:**

\[
\begin{align*}
f \bowtie g: f \bar{g} = g \bar{f}, & \quad f \bowtie g : f \bowtie g & \quad \text{and} & \quad f^\circ \bowtie g^\circ. \\
f: \quad a & \rightarrow a & a & \rightarrow a \\
& \downarrow & \downarrow & \\
b & \rightarrow b & c & \rightarrow c \\
g: \quad a & \rightarrow a & a & \rightarrow a \\
& \downarrow & \downarrow & \\
b & \rightarrow b & c & \rightarrow c \\
\end{align*}
\]

### Union of partial isos

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

\[
\begin{align*}
k: \quad a & \rightarrow a & a & \rightarrow a \\
& \downarrow & \downarrow & \\
b & \rightarrow b & c & \rightarrow c \\
\end{align*}
\]

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

\[
\text{if } s \leq t, \text{ then } s \leq \bigvee_{s \in S} s, \bigvee_{s \in S} s \leq t, \bigvee_{s \in S} s = \bigvee_{s \in S} \bar{s},
\]

\[
f \circ \left( \bigvee_{s \in S} s \right) = \bigvee_{s \in S} fs, \quad \left( \bigvee_{s \in S} s \right) \circ g = \bigvee_{s \in S} sg.
\]
Restriction and inverse category

Domain and partiality

Restriction category: \( f \mapsto \bar{f} \).

\[
\begin{align*}
    f \circ \bar{f} &= f, & f \circ \bar{g} &= \bar{g} \circ \bar{f}, & \bar{f} \circ \bar{g} &= f \circ \bar{g}, & \bar{h} \circ f &= f \circ h \circ f.
\end{align*}
\]

Partial inverse

Inverse category: \( f \mapsto \bar{f} \circ \bar{f} \). \( \circ \bar{f} = \bar{f} \) and \( f \circ \bar{f} = \bar{f} \circ f \).

Compatibility

Restriction compatible:

\[
\begin{align*}
    f \rightleftharpoons g : f \bar{g} &= g \bar{f}, & f \rightleftharpoons g : f \rightleftharpoons g \quad \text{and} \\
    f \circ \rightleftharpoons g &\circ \rightleftharpoons g\circ.
\end{align*}
\]

Union of partial isos

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

Join:

\[
\bigvee_{s \in S} s.
\]

if \( s \leq t \), then \( s \leq \bigvee_{s \in S} s \), \( \bigvee_{s \in S} s \leq t \), \( \overline{\bigvee_{s \in S} s} = \bigvee_{s \in S} \overline{s} \),

\[
\begin{align*}
    f \circ \left( \bigvee_{s \in S} s \right) &= \bigvee_{s \in S} f s, & \left( \bigvee_{s \in S} s \right) \circ g &= \bigvee_{s \in S} s g.
\end{align*}
\]

Id:

\[
\begin{align*}
    a &\rightarrow a, & b &\rightarrow b, & c &\rightarrow c
\end{align*}
\]
Example

$S$ a set. $\text{PId}_S$ category with one object $\ast$ 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

\[
[\Delta] = [a_1] \otimes \cdots \otimes [a_n] \text{ whenever } \Delta \vdash x_1 : a_1, \ldots, x_n : a_n.
\]

\[
[\Delta \vdash \text{inj}_I, \nu : a \oplus b] \doteq \iota_I \circ f, \text{ whenever } f = [\Delta \vdash \nu : a],
\]
and similarly for the right-projection.

If \( f = [\Delta_1 \vdash \nu_1 : a_1] \) and \( g = [\Delta_2 \vdash \nu_2 : a_2] \),
\[
[\Delta_1, \Delta_2 \vdash \langle \nu_1, \nu_2 \rangle : a_1 \otimes a_2] \doteq f \otimes g.
\]
An iso: \[ \{ v_1 \leftrightarrow v'_1 \mid \ldots \mid v_n \leftrightarrow v'_n \} : a \leftrightarrow b. \]
An iso: \( \{ v_1 \leftrightarrow v'_1 \mid \ldots \mid v_n \leftrightarrow v'_n \} : a \leftrightarrow b. \)

The situation:

\[
\begin{array}{c}
\Delta_i \\
\downarrow \omega \\
\n\end{array}
\quad
\begin{array}{c}
v_i \\
\downarrow \omega \\
v'_i \\
\end{array}
\quad
\begin{array}{c}
av \\
\rightarrow \\
\downarrow \omega \\
\rightarrow \\
\end{array}
\quad
\begin{array}{c}
b \\
\rightarrow \\
\downarrow \omega \\
\rightarrow \\
\end{array}
\]
An iso: \( \{ v_1 \leftrightarrow v'_1 \mid \ldots \mid v_n \leftrightarrow v'_n \} : a \leftrightarrow b. \)

The situation:

![Diagram of the situation](image)

What we’d like to build:

![Diagram of what we’d like to build](image)

Careful: this diagram does not commute!
Lemma

If $v_1 \perp v_2$ and $v'_1 \perp v'_2$, then $[v'_1] \circ [v_1]^\circ \simeq [v'_2] \circ [v_2]^\circ$. 
Lemma

If $v_1 \perp v_2$ and $v_1' \perp v_2'$, then $[v_1'] \circ [v_1]^\circ \simeq [v_2'] \circ [v_2]^\circ$.

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

\[ [\{ \vdash \omega \{ v_1 \leftrightarrow v'_1 \mid v_2 \leftrightarrow v'_2 \ldots \} : a \leftrightarrow b \} = \bigvee_i [v'_i] \circ [v_i]^\circ : [a] \rightarrow [b] ] . \]
What we would like to have thus

The language is based is pattern-matching:

\[ \vdash_\omega \{ \mid v_1 \leftrightarrow v'_1 \mid v_2 \leftrightarrow v'_2 \ldots \} \]

decidable which pattern a term \( u \) fits.

It should also be in the denotational semantics:

\[
\left( ([v_1] \circ [v_1]) \land ([v_2] \circ [v_2]) \right) \circ [u] = [v'_i] \circ [v_i] \circ [u]
\]

for some \( i \).
The language is based is pattern-matching:

$$\vdash_\omega \{ \quad \nu_1 \leftrightarrow \nu_1' \mid \nu_2 \leftrightarrow \nu_2' \quad \cdots \}$$

decidable which pattern a term \( u \) fits.

It should also be in the denotational semantics:

$$((\nu_1' \circ [\nu_1]) \lor (\nu_2' \circ [\nu_2])) \circ [u] = [\nu_i'] \circ [\nu_i] \circ [u]$$

for some \( i \).

Usually, it relies on disjointness (\( \oplus \)).
Pattern-matching and 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 \((f \lor g)h\), we want \(h\) to choose:

\[ fh \lor gh = fh \text{ or } fh \lor gh = gh \]

A join inverse category with this property will be called \textit{pattern-matching} category. With disjointness \((\oplus)\), rather \(fh = 0\) or \(gh = 0\).
The origin of consistency

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

\[
fh \lor gh = fh \text{ or } fh \lor gh = gh
\]

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 \(fh \lor gh\) not decomposable.

For this, we need \(h\) to be *not decomposable* either.
The origin of consistency

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

\[ fh \lor gh = fh \text{ or } fh \lor gh = gh \]

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

The idea: translate it as a notion of \textit{non decomposability}, to have \(fh \lor gh\) not decomposable.

For this, we need \(h\) to be \textit{not decomposable} either.

Consistency: for any morphism \(k\), if \(h\) is \textit{non decomposable}, so is \(kh\).

A join inverse category with this property will be called \textit{consistent} category.
In \((f \lor g)h\), we want \(h\) to choose:

\[ fh \lor gh = fh \text{ or } fh \lor gh = gh \]

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

\textbf{The idea:} translate it as a notion of \textit{non decomposability}, to have \(fh \lor gh\) not decomposable.

For this, we need \(h\) to be \textit{not decomposable} either.

\textbf{Consistency:} for any morphism \(k\), if \(h\) is \textit{non decomposable}, so is \(kh\).

A join inverse category with this property will be called \textit{consistent} category.

Consistency implies pattern-matching.
The origin of consistency

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

\[
fh \lor gh = fh \text{ or } fh \lor gh = gh
\]

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

**The idea:** translate it as a notion of \textit{non decomposability}, to have \(fh \lor gh\) not decomposable.

For this, we need \(h\) to be \textit{not decomposable} either.

**Consistency:** for any morphism \(k\), if \(h\) is \textit{non decomposable}, so is \(kh\).

A join inverse category with this property will be called \textit{consistent} category.

Consistency implies pattern-matching.

What about the other way around?
The different kinds

**Definition (Strongly non decomposable)**

\[
0 \quad \longrightarrow \quad m
\]

Thus \( m = u \lor v \Rightarrow u = 0 \) or \( v = 0 \)
The different kinds

<table>
<thead>
<tr>
<th>Definition (Strongly non decomposable)</th>
<th>0 —— m</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thus $m = u \lor v \Rightarrow u = 0$ or $v = 0$</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Definition (Linearly non decomposable)</th>
<th>0 —····— f — m</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thus $m = u \lor v \Rightarrow u \leq v$ or $v \leq u$</td>
<td></td>
</tr>
</tbody>
</table>
The different kinds

**Definition (Strongly non decomposable)**

\[ 0 \rightarrow m \]

Thus \( m = u \vee v \Rightarrow u = 0 \) or \( v = 0 \)

**Definition (Linearly non decomposable)**

\[ 0 \rightarrow \cdots \rightarrow f \rightarrow m \]

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)**

\[ 0 \rightarrow m \]

Thus \( m = u \lor v \Rightarrow u = 0 \) or \( v = 0 \)

**Definition (Linearly non decomposable)**

\[ 0 \rightarrow \cdots \rightarrow f \rightarrow m \]

Thus \( m = u \lor v \Rightarrow u \leq v \) or \( v \leq u \)

**Definition (Weakly non decomposable)**

\[ \begin{array}{c}
  0 \\
  \quad f \\
  \quad m \\
  \quad h
\end{array} \]

Thus \( m = u \lor v \Rightarrow u = m \) or \( v = m \)

**strongly \( \Rightarrow \) linearly \( \Rightarrow \) weakly**
The results

**Theorem**

A weakly pattern-matching category is weakly consistent.
### Theorem

A weakly pattern-matching category is weakly consistent.

### Theorem

A join inverse category is strongly consistent.
### The results

<table>
<thead>
<tr>
<th>Theorem</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A weakly pattern-matching category is weakly consistent.</td>
<td></td>
</tr>
<tr>
<td>Theorem</td>
<td></td>
</tr>
<tr>
<td>A join inverse category is strongly consistent.</td>
<td></td>
</tr>
<tr>
<td>Theorem</td>
<td></td>
</tr>
<tr>
<td>A join inverse category is linearly consistent.</td>
<td></td>
</tr>
</tbody>
</table>
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 $\text{PId}_S$ 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 PlId$_S$ 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 $\Pi \text{Id}_S$ as a sound and adequate denotation.
Going further

**Recursion**

$C$ a join inverse category.

For any $A$, $B$ objects of $C$, $\text{Hom}_C(A, B)$ is a DCPO.

**Quantum programming language**

Patterns are vectors in a basis of a Hilbert space.

Work in progress.
Thank you