Reversible effects as inverse arrows

Chris Heunen   Robin Kaarsgaard   Martti Karvonen

THE UNIVERSITY of EDINBURGH informatics

UNIVERSITY OF COPENHAGEN

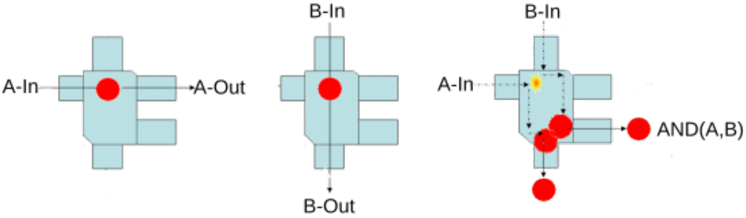
Outline

- Arrows add non-functional side-effects to functional languages
- Reversible languages take semantics in inverse categories

Arrows	categories
Inverse arrows	inverse categories
dagger arrows	dagger categories

- Many examples of inverse arrows

Reversible and invertible programming



Functional languages not stateful by definition, easing reversibility

(e.g. Theseus, RFun)

Monads

$$\text{return} : X \to M\ X$$
$$(\gg\!=) : M\ X \to (X \to M\ Y) \to M\ Y$$

such that:

$$\begin{aligned}
\text{return}\, x \gg\!= f &= f x \\
m \gg\!= \text{return} &= m \\
(m \gg\!= f) \gg\!= g &= m \gg\!= (\lambda x. f x \gg\!= g)
\end{aligned}$$

Monads

$$\text{return} : X \to M\ X$$
$$(\gg=) : M\ X \to (X \to M\ Y) \to M\ Y$$

think:

- $M\ X$ is (effectful) computation of type X
- return x is constant computation
- composition $\gg=$ should behave

Arrows

$$
\begin{aligned}
\mathrm{arr} &: (X \to Y) \to A\ X\ Y \\
(\ggg) &: A\ X\ Y \to A\ Y\ Z \to A\ X\ Z \\
\mathrm{first}_{X,Y,Z} &: A\ X\ Y \to A\ (X \otimes Z)\ (Y \otimes Z)
\end{aligned}
$$

such that:

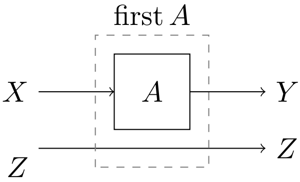
$$
\begin{aligned}
(a \ggg b) \ggg c &= a \ggg (b \ggg c) \\
\mathrm{arr}(g \circ f) &= \mathrm{arr}\, f \ggg \mathrm{arr}\, g \\
\mathrm{arr}\,\mathrm{id} \ggg a = {} & a = a \ggg \mathrm{arr}\,\mathrm{id} \\
\mathrm{first}_{X,Y,I}\, a \ggg \mathrm{arr}\, \rho_Y &= \mathrm{arr}\, \rho_X \ggg a \\
\mathrm{first}_{X,Y,Z}\, a \ggg \mathrm{arr}(\mathrm{id}_Y \otimes f) &= \mathrm{arr}(\mathrm{id}_X \otimes f) \ggg \mathrm{first}_{X,Y,Z}\, a \\
(\mathrm{first}_{X,Y,Z \otimes V}\, a) \ggg \mathrm{arr}\, \alpha_{Y,Z,V} &= \mathrm{arr}\, \alpha_{X,Z,V} \ggg \mathrm{first}(\mathrm{first}\, a) \\
\mathrm{first}(\mathrm{arr}\, f) &= \mathrm{arr}(f \otimes \mathrm{id}) \\
\mathrm{first}(a \ggg b) &= (\mathrm{first}\, a) \ggg (\mathrm{first}\, b)
\end{aligned}
$$

Arrows

$$\text{arr} : (X \to Y) \to A\ X\ Y$$
$$(\ggg) : A\ X\ Y \to A\ Y\ Z \to A\ X\ Z$$
$$\text{first}_{X,Y,Z} : A\ X\ Y \to A\ (X \otimes Z)\ (Y \otimes Z)$$

think:

- $A\ X\ Y$ is type of effectful computations from X to Y
- arr makes pure computation effectful
- composition $\ggg$ behaves properly
- first lets effectful computations interact with environment



Dagger and inverse arrows

$$\mathrm{inv} : A\ X\ Y \to A\ Y\ X$$

such that:

$$\begin{aligned}
\mathrm{inv}(\mathrm{inv}\, a) &= a \\
\mathrm{inv}\, a \ggg \mathrm{inv}\, b &= \mathrm{inv}(b \ggg a) \\
\mathrm{arr}(f^{\dagger}) &= \mathrm{inv}(\mathrm{arr}\, f) \\
\mathrm{inv}(\mathrm{first}\, a) &= \mathrm{first}(\mathrm{inv}\, a)
\end{aligned}$$

$$\begin{aligned}
(a \ggg \mathrm{inv}\, a) \ggg a &= a \\
(a \ggg \mathrm{inv}\, a) \ggg (b \ggg \mathrm{inv}\, b) &= (b \ggg \mathrm{inv}\, b) \ggg (a \ggg \mathrm{inv}\, a)
\end{aligned}$$

Dagger and inverse arrows

$$\mathrm{inv} : A\ X\ Y \to A\ Y\ X$$

think:

- inv turns effectful computations around
- cooperates with pure computations and environments
- inv a 'undoes' a

Example: reversible state

$\textbf{type}\ \mathit{State}\ S\ X\ Y = X \to (S \multimap (X \otimes S))$
$\textbf{type}\ \mathit{RState}\ S\ X\ Y = X \otimes S \leftrightarrow Y \otimes S$

$\textbf{instance}\ \mathit{Arrow}\ (\mathit{RState}\ S)\ \textbf{where}$

$$\begin{aligned} \mathit{arr}\ f\ (x, s) &= (f\ x, s) \\ (a \ggg b)(x, s) &= b(a(x, s)) \\ \mathit{first}\ a\ ((x, z), s) &= \textbf{let}\ (x', s') = a(x, s)\ \textbf{in}\ ((x', z), s') \end{aligned}$$

$\textbf{instance}\ \mathit{InverseArrow}\ (\mathit{RStateS})\ \textbf{where}$

$$\mathit{inv}\ a\ (y, s) = a^{\dagger}(y, s)$$

Example: reversible state

$$\textbf{type}\ \mathit{State}\ S\ X\ Y = X \to (S \multimap (X \otimes S))$$
$$\textbf{type}\ \mathit{RState}\ S\ X\ Y = X \otimes S \leftrightarrow Y \otimes S$$

$$\begin{array}{l} \textbf{instance}\ \mathit{Arrow}\ (\mathit{RState}\ S)\ \textbf{where} \\ \quad \mathit{arr}\ f\ (x, s) = (f\ x, s) \\ \quad (a \ggg b)(x, s) = b(a(x, s)) \\ \quad \mathit{first}\ a\ ((x, z), s) = \textbf{let}\ (x', s') = a(x, s)\ \textbf{in}\ ((x', z), s') \end{array}$$

$$\begin{array}{l} \textbf{instance}\ \mathit{InverseArrow}\ (\mathit{RStateS})\ \textbf{where} \\ \quad \mathit{inv}\ a\ (y, s) = a^{\dagger}(y, s) \end{array}$$

$$\begin{array}{ll} \mathit{get} & :\ \mathit{RState}\ S\ X\ (X \otimes S) \\ \mathit{get}\ (x, s) & = ((x, s), s) \end{array}$$

$$\begin{array}{ll} \mathit{update} & :\ (S \leftrightarrow S) \to \mathit{RState}\ S\ X\ X \\ \mathit{update}\ f\ (x, s) & = (x, f\ s) \end{array}$$

Example: rewriter

$$
\begin{aligned}
&\textbf{class } \mathit{Group}\ G \textbf{ where}\\
&\quad \mathit{gunit} : G\\
&\quad \mathit{gmul} : G \to (G \leftrightarrow G)\\
&\quad \mathit{ginv} \;\, : G \leftrightarrow G
\end{aligned}
$$

$$\textbf{type } \mathit{Rewriter}\ G\ X\ Y = X \otimes G \leftrightarrow Y \otimes G$$

Example: rewriter

$$\textbf{class}\ \mathit{Group}\ G\ \textbf{where}$$
$$\mathit{gunit} : G$$
$$\mathit{gmul} : G \to (G \leftrightarrow G)$$
$$\mathit{ginv} : G \leftrightarrow G$$

$$\textbf{type}\ \mathit{Rewriter}\ G\ X\ Y = X \otimes G \leftrightarrow Y \otimes G$$

$$\mathit{rewrite} : G \to \mathit{Rewriter}\ G\ X\ X$$
$$\mathit{rewrite}\ a\ (x, b) = (x, \mathit{gmul}\ a\ b)$$

Example: vector transformations

$$\textbf{type } \textit{Vector } X \; Y = [X] \leftrightarrow [Y]$$

$$\begin{aligned}
&\textbf{instance } \textit{Arrow } (\textit{Vector}) \textbf{ where} \\
&\quad \textit{arr } f \; xs = \textit{map } f \; xs \\
&\quad (a \ggg b) \; xs = b \; (a \; xs) \\
&\quad \textit{first } a \; ps = \textbf{let } (xs, zs) = \textit{zip}^{\dagger} \; ps \textbf{ in } \textit{zip } (a \; xs, zs) \\
&\textbf{instance } \textit{InverseArrow } (\textit{Vector}) \textbf{ where} \\
&\quad \textit{inv } a \; ys = a^{\dagger} \; ys
\end{aligned}$$

$$\begin{aligned}
\textit{map} &: (a \leftrightarrow b) \to ([a] \leftrightarrow [b]) \\
\textit{map } f \; [] &= [] \\
\textit{map } f \; (x{::}xs) &= (f \; x){::}(\textit{map } f \; xs)
\end{aligned}$$

$$\begin{aligned}
\textit{zip} &: ([a], [b]) \leftrightarrow [(a, b)] \\
\textit{zip}([], []) &= [] \\
\textit{zip}(x{::}xs, y{::}ys) &= (x, y){::}(\textit{zip}(xs, ys))
\end{aligned}$$

Example: serialization

$$serialize : X \leftrightarrow Serialized\ X$$

$$\textbf{type}\ Serializer\ X\ Y = X \leftrightarrow Serialized\ Y$$

$$\begin{array}{l}
\textbf{instance}\ Arrow\ (Serializer)\ \textbf{where} \\
\quad arr\ f\ x \qquad\quad = serialize\ (f\ x) \\
\quad (a \ggg b)\ x\ \ = b\ (serialize^{\dagger}(a\ x)) \\
\quad first\ a\ (x, z) = serialize(serialize^{\dagger}(a\ x), z)
\end{array}$$

$$\begin{array}{l}
\textbf{instance}\ InverseArrow\ (Serializer)\ \textbf{where} \\
\quad inv\ a\ y = serialize(a^{\dagger}(serialize\ y))
\end{array}$$

Example: error handling

type $Error\ E\ X\ Y = X \oplus E \leftrightarrow Y \oplus E$

instance $WeakArrow\ (Error\ E)$ **where**

$$
\begin{aligned}
&arr\ f\ (InL\ x) = InL\ (f\ x) \\
&arr\ f\ (InR\ e) = InR\ e \\
&(a \ggg b)\ x \quad = b\ (a\ x)
\end{aligned}
$$

instance $InverseWeakArrow\ (Error\ E)$ **where**

$$inv\ a\ y = a^{\dagger}\ y$$

Example: error handling

$$\textbf{type}\ \mathit{Error}\ E\ X\ Y = X \oplus E \leftrightarrow Y \oplus E$$

$$\begin{array}{l}
\textbf{instance}\ \mathit{WeakArrow}\ (\mathit{Error}\ E)\ \textbf{where} \\
\quad \mathit{arr}\ f\ (\mathit{InL}\ x) = \mathit{InL}\ (f\ x) \\
\quad \mathit{arr}\ f\ (\mathit{InR}\ e) = \mathit{InR}\ e \\
\quad (a \ggg b)\ x \quad = b\ (a\ x)
\end{array}$$

$$\begin{array}{l}
\textbf{instance}\ \mathit{InverseWeakArrow}\ (\mathit{Error}\ E)\ \textbf{where} \\
\quad \mathit{inv}\ a\ y = a^{\dagger}\ y
\end{array}$$

$$\begin{array}{l}
\mathit{raise} \qquad\quad : \ (X \leftrightarrow E) \to (E \leftrightarrow E \oplus E) \to \mathit{Error}\ E\ X\ Y \\
\mathit{raise}\ f\ p\ x = \mathit{InR}\ (p^{\dagger}\ (\mathit{arr}\ f\ x)))
\end{array}$$

Example: superoperators

Quantum physical maps $f\colon X \to Y$ don't just take states to states.
Must respect entanglement with environment:
so $f \otimes \mathrm{id}\colon X \otimes E \to Y \otimes E$ takes states to states.
Leads to CPM construction, *not* a monad, but dagger arrow:

$$
\begin{aligned}
A\ X\ Y &= \{\text{ completely positive maps } X^* \otimes X \to Y^* \otimes Y\} \\
\mathrm{arr}\ \ f &= f_* \otimes f \\
a \ggg b &= b \circ a \\
\mathrm{first}_{X,Y,Z}\ \ a &= a \otimes \mathrm{id}_{Z^* \otimes Z} \\
\mathrm{inv}\ \ a &= a^\dagger
\end{aligned}
$$

Examples: many more

- Pure functions: program inverter
- Dagger Frobenius monads, restriction monads
- Control flow: ArrowChoice
- Computation in context:

  $$\textbf{type}\ \mathit{Reader}\ C\ X\ Y = X \otimes C \leftrightarrow Y \otimes C$$

- Information effects [James & Sabry]: irreversible computation in pure reversible setting with inverse arrow for implicit communication with heap and garbage dump
- Reversible IO: must be built into programming language
- Reversible recursion: type separating non/terminating functions

Reversible categories

- In dagger category, $X \xrightarrow{f} Y$ has partner $X \xleftarrow{f^\dagger} Y$ with $f^{\dagger\dagger} = f$
- In inverse category, moreover:
  - $f \circ f^\dagger \circ f = f$
  - $f^\dagger \circ f \circ g^\dagger \circ g = g^\dagger \circ g \circ f^\dagger \circ f$
- If monoidal, also want $(f \otimes g)^\dagger = f^\dagger \otimes g^\dagger$

Examples:

- Any groupoid
- Sets and relations
- Sets and partial injections (universal)
- Hilbert spaces

Arrows, categorically

- Monoid: object M with maps $M \otimes M \to M \leftarrow I$ satisfying laws
- Monad on $\mathbf{C}$ is monoid in endofunctor category $[\mathbf{C}, \mathbf{C}]$
- Profunctors $\mathbf{C}^{\mathrm{op}} \times \mathbf{C} \to \mathbf{Set}$ are monoidal under

$$(F \otimes G)(X, Z) = \int^{Y} F(X, Y) \times G(Y, Z)$$

Theorem (2006): Arrow = strong monoid in $[\mathbf{C}^{\mathrm{op}} \times \mathbf{C}, \mathbf{Set}]$

Dagger arrows, categorically

- Involutive monoidal category has functor $\overline{(-)}\colon \mathbf{C} \to \mathbf{C}$ with $\overline{\overline{f}} = f$ and coherent natural $\overline{X} \otimes \overline{Y} \simeq \overline{Y \otimes X}$
- Involutive monoid is monoid with monoid map $i\colon \overline{M} \to M$ satisfying $i \circ \overline{i} = \mathrm{id}$

Lemma: if $\mathbf{C}$ is dagger, then $[\mathbf{C}^{\mathrm{op}} \times \mathbf{C}, \mathbf{Set}]$ is involutive

$$\overline{F}(f, g) = F(g^\dagger, f^\dagger) \qquad \overline{\alpha}_{X,Y} = \alpha_{Y,X}$$

Theorem: Dagger arrow = involutive monoid in $[\mathbf{C}^{\mathrm{op}} \times \mathbf{C}, \mathbf{Set}]$

Inverse arrow makes three additional diagrams commute

Conclusion

- definition of inverse arrow
- supports many examples
- has clean categorical structure

- informs sound reversible programming language design
- (un)do-notation

References

- *"Arrows, like Monads, are Monoids"*
  C. Heunen, B. Jacobs, MFPS, 2006
- *"Categorical semantics for Arrows"*
  B. Jacobs, C. Heunen, I. Hasuo, Journal of Functional Programming, 2009
- *"Reversible monadic programming"*
  C. Heunen, M. Karvonen, MFPS, 2015
- *"Monads on dagger categories"*
  C. Heunen, M. Karvonen, Theory and Applications of Categories, 2016
- *"Join inverse categories as models of reversible recursion"*
  H. B. Axelsen, R. Kaarsgaard, FoSSaCS, 2016
- *"Join inverse categories and reversible recursion"*
  R. Kaarsgaard, H. B. Axelsen, R. Gluck, Journal of Logical and Algebraic Methods in Programming, 2017

Involutive monoidal categories

Functor $\overline{(-)}\colon \mathbf{C} \to \mathbf{C}$ with $\overline{\overline{f}} = f$, coherent natural $\overline{X} \otimes \overline{Y} \simeq \overline{Y \otimes X}$:

$$
\begin{array}{ccc}
\overline{X} \otimes (\overline{Y} \otimes \overline{Z}) & \xrightarrow{\ \alpha\ } & (\overline{X} \otimes \overline{Y}) \otimes \overline{Z} \\
{\scriptstyle \mathrm{id} \otimes \chi}\big\downarrow & & \big\downarrow{\scriptstyle \chi \otimes \mathrm{id}} \\
\overline{X} \otimes \overline{Z \otimes Y} & & \overline{Y \otimes X} \otimes \overline{Z} \\
{\scriptstyle \alpha}\big\downarrow & & \big\downarrow{\scriptstyle \chi} \\
\overline{(Z \otimes Y) \otimes X} & \xleftarrow[\ \overline{\alpha}\ ]{} & \overline{Z \otimes (Y \otimes X)}
\end{array}
\qquad\qquad
\begin{array}{ccc}
\overline{\overline{X}} \otimes \overline{\overline{Y}} & \xrightarrow{\ \chi\ } & \overline{\overline{Y} \otimes \overline{X}} \\
{\scriptstyle \mathrm{id}}\big\downarrow & & \big\downarrow{\scriptstyle \overline{\chi}} \\
X \otimes Y & \xrightarrow[\ \mathrm{id}\ ]{} & \overline{\overline{X \otimes Y}}
\end{array}
$$

Inverse arrow laws

$$L\colon [\mathbf{C}^{\mathrm{op}} \times \mathbf{C}, \mathbf{Set}] \to [\mathbf{C}^{\mathrm{op}} \times \mathbf{C}, \mathbf{Set}]$$
$$LM(X, Y) = M(X, X)$$
$$LM(f, g) = f^\dagger \circ (-) \circ f$$

For M involutive monoid:

$$L^+M(X, Y) = \{a^\dagger \circ a \in M(X, X) \mid a \in M(X, Z) \text{ for some } Z\}$$

Inverse arrow law 1

$$g^\dagger \circ g \circ b^\dagger \circ b = b^\dagger \circ b \circ g^\dagger \circ g \text{ for pure } g$$

$$\begin{aligned} L^+(\text{hom}) \times LM &\to LM \\ (g^\dagger \circ g, a) &\mapsto g^\dagger \circ g \circ a \end{aligned}$$

$$\begin{array}{ccccc} L^+M \times L^+(\text{hom}) & & \longrightarrow & & LM \times L^+(\text{hom}) \\ \sigma\downarrow & & & & \downarrow \\ L^+(\text{hom}) \times L^+M & \longrightarrow & L^+(\text{hom}) \times LM & \longrightarrow & LM \end{array}$$

Inverse arrow law 2

$$a^\dagger \circ a \circ b^\dagger \circ b = b^\dagger \circ b \circ a^\dagger \circ a$$

$$\begin{aligned} L^+M \times L^+M &\to LM \\ (a^\dagger \circ a, b^\dagger \circ b) &\mapsto a^\dagger \circ a \circ b^\dagger \circ b \end{aligned}$$

$$\begin{array}{ccc} L^+M \times L^+M & \xrightarrow{\quad\sigma\quad} & L^+M \times L^+M \\ & \searrow & \downarrow \\ & & LM \end{array}$$

Inverse arrow law 3

$$a \circ a^\dagger \circ a = a$$

$$D_M \hookrightarrow M \times \overline{M} \times M$$

$$D_M(X, Y) = \{(a, a^\dagger, a) \mid a \in M(X, Y)\}$$