Factorisation systems for logical relations and monadic lifting in type-and-effect system semantics

Ohad Kammar
University of Oxford

Dylan McDermott
University of Cambridge

Type-and-effect systems

effect — type

- Caching: $\Gamma \vdash_{\{\mathsf{write}\}} M : \mathrm{int}$

$$(M, M) \qquad \equiv \qquad \mathsf{let}\ x = M\ \mathsf{in}\ (x, x)$$

- Swapping: $\Gamma \vdash_{\{\mathsf{read}\}} M : 1,\ \Gamma \vdash_{\{\mathsf{read}\}} K : 1$

$$M; K \qquad \equiv \qquad K; M$$

semantic justification?

Type-and-effect systems

ground types

parameter type arity

signature operation name

$$\Sigma = \{\mathsf{read} : 1 \to V,\ \mathsf{write} : V \to 1,\ \mathsf{raise} : \mathsf{exn} \to 0, \dots\}$$

$$\begin{aligned}
M, N \ ::= \ & c \mid x \mid () \mid (M, N) \mid \mathsf{fst}\, M \mid \mathsf{snd}\, M \mid \mathsf{elim}_0\, M \\
& \mid \mathsf{inl}\, M \mid \mathsf{inr}\, M \mid \mathsf{match}\ M\ \mathsf{with}\ \{\mathsf{inl}\, x.\, N_1, \mathsf{inr}\, y.\, N_2\} \\
& \mid \lambda x.\, M \mid M\, N \mid \mathsf{let}\ x = M\ \mathsf{in}\ N \\
& \mid \mathsf{op}\, M & \mathsf{op} \in \Sigma \\
A, B ::= \ & b \mid 1 \mid A \times B \mid 0 \mid A + B \\
& \mid A \xrightarrow{\varepsilon} B & \varepsilon \subseteq_{\mathsf{fin}} \Sigma
\end{aligned}$$

Erasure:

$$\underline{\mathrm{int} \xrightarrow{\{\mathsf{write}\}} 1} := \mathrm{int} \to 1 \qquad \underline{x_1 : A_1, \dots, x_n : A_n} := x_1 : \underline{A_1}, \dots, x_n : \underline{A_n}$$

$$\Gamma \vdash_\varepsilon M : A \qquad \Rightarrow \qquad \underline{\Gamma} \vdash M : \underline{A}$$

refined judgment unrefined judgment

Type-and-effect systems

$$
\begin{tikzcd}
 & T_{\{\mathsf{read}\}} \arrow[r] \arrow[dr] & T_{\{\mathsf{read},\mathsf{write}\}} \arrow[dr] & \\
T_{\emptyset} \arrow[ur] \arrow[r] \arrow[dr] & T_{\{\mathsf{write}\}} \arrow[ur] \arrow[dr] & T_{\{\mathsf{read},\mathsf{raise}\}} \arrow[r] & T_{\{\mathsf{read},\mathsf{write},\mathsf{raise}\}} \\
 & T_{\{\mathsf{raise}\}} \arrow[r] \arrow[ur] & T_{\{\mathsf{write},\mathsf{raise}\}} \arrow[ur] &
\end{tikzcd}
$$

strong monad

Refined models:

- ε-monad T_ε for each $\varepsilon \subseteq \Sigma$

$[\![op]\!] : [\![G]\!] \to T_\varepsilon\,[\![G']\!]$

for each

$(op : G \to G') \in \varepsilon$

- ε-monad morphism $T_\varepsilon \to T_{\varepsilon'}$ for each $\varepsilon \subseteq \varepsilon'$

$$[\![\Gamma \vdash_\varepsilon M : A]\!] : [\![\Gamma]\!] \to T_\varepsilon\,[\![A]\!]$$

Type-and-effect systems

$$\left[\!\!\left[f : 1 \xrightarrow{\{\mathsf{write}\}} \mathrm{int} \vdash_{\{\mathsf{write}\}} (f(), f()) : \mathrm{int} \times \mathrm{int} \right]\!\!\right]$$
$$= \left[\!\!\left[f : 1 \xrightarrow{\{\mathsf{write}\}} \mathrm{int} \vdash_{\{\mathsf{write}\}} \mathsf{let}\ x = f()\ \mathsf{in}\ (x, x) : \mathrm{int} \times \mathrm{int} \right]\!\!\right]$$

but

$$[\![f : 1 \to \mathrm{int} \vdash (f(), f()) : \mathrm{int} \times \mathrm{int}]\!]$$
$$\neq [\![f : 1 \to \mathrm{int} \vdash \mathsf{let}\ x = f()\ \mathsf{in}\ (x, x) : \mathrm{int} \times \mathrm{int}]\!]$$

Constructing refined semantics

$$\begin{tikzcd}
 & & T_{\{\mathsf{read}\}} \arrow[r, tail] \arrow[dr, tail] & T_{\{\mathsf{read},\mathsf{write}\}} \arrow[dr, tail] & \\
T \arrow[r, mapsto] & T_{\emptyset} \arrow[ur, tail] \arrow[r, tail] \arrow[dr, tail] & T_{\{\mathsf{write}\}} \arrow[ur, tail] \arrow[dr, tail] & T_{\{\mathsf{read},\mathsf{raise}\}} \arrow[r, tail] & T_{\{\mathsf{read},\mathsf{write},\mathsf{raise}\}} \\
 & & T_{\{\mathsf{raise}\}} \arrow[r, tail] \arrow[ur, tail] & T_{\{\mathsf{write},\mathsf{raise}\}} \arrow[ur, tail] &
\end{tikzcd}$$

Contribution: construction of refined semantics

- $T \mapsto T_\varepsilon$ via monad factorisation
- Sound and complete:

$$[\![\Gamma \vdash_\varepsilon M : G]\!] = [\![\Gamma \vdash_\varepsilon K : G]\!] \quad \Leftrightarrow \quad [\![\Gamma \vdash M : G]\!] = [\![\Gamma \vdash K : G]\!]$$

ground context ground type

Factoring monad morphisms

A *factorisation system* consists of:

"epis"

- A class $\mathcal{E}$ of morphisms $e : X \twoheadrightarrow Y$
- A class $\mathcal{M}$ of morphisms $n : X \rightarrowtail Y$

"monos"

such that:

- Every morphism can be factored:

$$\begin{tikzcd} X \arrow[rr, "f"] \arrow[dr, two heads, "e"'] & \arrow[d, phantom, "="] & Y \\ & Z \arrow[ur, tail, "n"'] & \end{tikzcd}$$

- $\mathcal{E}$, $\mathcal{M}$ closed under composition, contain isos
- $\mathcal{E}$ is left orthogonal to $\mathcal{M}$:

$$\begin{tikzcd} W \arrow[r, two heads, "e"] \arrow[d] \arrow[dr, phantom, "="] & X \arrow[d] \\ Y \arrow[r, tail, "n"'] & Z \end{tikzcd} \quad \Rightarrow \quad \begin{tikzcd} W \arrow[r, two heads, "e"] \arrow[d] & X \arrow[d] \arrow[dl, dashed] \\ Y \arrow[r, tail, "n"'] & Z \end{tikzcd}$$

Factoring monad morphisms

Examples of factorisation systems

- **Set**:

  (surjection, injection)

- ω**Cpo**:

  $n\,x \leq n\,y \Rightarrow x \leq y$

  (dense functions, full functions)

  ω-chain-closure($e[domain]$) $=$ *codomain*

- Functor categories:

  (componentwise $\mathcal{E}$, componentwise $\mathcal{M}$)

Factoring monad morphisms

Theorem

F-monad morphisms $m : S \to T$ factor componentwise:

$$\begin{tikzcd} SX \arrow[rr, "m_X"] \arrow[dr, two heads, "e_X"'] & \arrow[d, phantom, "="] & TX \\ & RX \arrow[ur, tail, "n_X"'] & \end{tikzcd}$$

$e \in \mathcal{E} \implies Se, Fe \in \mathcal{E}$

If $\mathcal{E}$ is closed under S, products, and F then:

- *R is an F-monad*
- *e and n are F-monad morphisms*

$e_1, e_2 \in \mathcal{E} \implies e_1 \times e_2 \in \mathcal{E}$

Pay as you go — drop:

- F-algebra structure
- strength
- monad laws

F-monad:

$$\begin{tikzcd} X \times F(TY) \arrow[r, "\mathrm{str}^F"] \arrow[d, "\mathrm{id} \times \beta"'] & F(X \times TY) \arrow[r, "F\mathrm{str}^T"] \arrow[d, phantom, "="] & F(T(X \times Y)) \arrow[d, "\beta"] \\ X \times TY \arrow[rr, "\mathrm{str}^T"'] & & T(X \times Y) \end{tikzcd}$$

Canonical S and m

Given $\varepsilon \subseteq \Sigma$, define

$$F_\varepsilon := \sum_{(\mathrm{op}\colon G \to G') \in \varepsilon} [\![G]\!] \times ([\![G']\!] \Rightarrow (-))$$

$$S_\varepsilon := \text{the free } F_\varepsilon\text{-monad}$$

Apply factorisation theorem:

Assuming $\mathcal{C}$ is:

- locally presentable; and
- bicartesian closed;

$$\begin{tikzcd} S_\varepsilon X \arrow[rr, dashed] \arrow[dr, two heads] & \arrow[d, phantom, "="] & TX \\ & T_\varepsilon X \arrow[ur, tail] & \end{tikzcd}$$

$$T_{\varepsilon \subseteq \varepsilon'} : T_\varepsilon \rightarrowtail T_{\varepsilon'}$$

factorisation system
functoriality

unrefined model $\mapsto$ refined model

Examples

In Set: (for $|V|, |R| > 1$)

- $T = V \Rightarrow V \times -$ (global state [Kammar'14])
  $T = (V \Rightarrow - \Rightarrow R) \Rightarrow V \Rightarrow R$ (global state+continuations)

$$\begin{tikzcd}
 & T_{\{\mathrm{read}\}} \cong V \Rightarrow - \arrow[drr, tail] & & S_{\{\mathrm{read}\}} \arrow[ll, two heads] \arrow[d] \\
T_{\emptyset} \cong \mathrm{Id} \arrow[ur, tail] \arrow[dr, tail] & & & T_{\{\mathrm{read,write}\}} \cong V \Rightarrow V \times - \\
 & T_{\{\mathrm{write}\}} \cong (1 + V) \times - \arrow[urr, tail] & & S_{\{\mathrm{write}\}} \arrow[ll, two heads] \arrow[u]
\end{tikzcd}$$

Examples

In ω**Cpo**:

- $T = (- + E)_\perp$ (exceptions+partiality [Kammar-Plotkin'12])

$$T_{\{\mathsf{diverge}\}} \cong (-)_\perp$$

Examples

finite sets and injections

$[\mathbb{I}, \mathbf{Set}]$

- $T X n = V^n \Rightarrow \int^{m \in \mathbb{I}} \mathbb{I}(n, m) \times V^m \times X m$ (local state)

commutative monad

$$\begin{tikzcd}
T_{\{\mathrm{read},\mathrm{write}\}} X n \;\cong\; V^n \Rightarrow V^n \times X n \arrow[dr, tail] & \\
 & T_{\{\mathrm{read},\mathrm{write},\mathrm{alloc}\}} \\
T_{\{\mathrm{alloc}\}} X n \;\cong\; \int^{m \in \mathbb{I}} \mathbb{I}(n, m) \times V^{m-n} \times X m \arrow[ur, tail] &
\end{tikzcd}$$

Goal (Completeness)

ground

$$[\![\Gamma \vdash_\varepsilon M : G]\!] = [\![\Gamma \vdash_\varepsilon N : G]\!] \quad \Leftrightarrow \quad [\![\Gamma \vdash M : G]\!] = [\![\Gamma \vdash N : G]\!]$$

Strategy:

- construct **LogRel**: category for logical relations
- lift (T_ε, T) to **LogRel**
- Interpret programs in refined **LogRel** model

$$\begin{tikzcd}
\mathbf{LogRel} \arrow[loop above, "\dot{T}_\varepsilon"] \arrow[d, "p"] \\
\mathcal{C} \times \mathcal{C} \arrow[loop left, "T_\varepsilon"] \arrow[loop right, "T"]
\end{tikzcd}$$

predicates

Definition (Fibration for logical relations [Katsumata '13])

For a bi-ccc $\mathcal{C}$:

Fibration for logical relations $p : \mathcal{D} \to \mathcal{C}$: faithful functor such that:

- p is a bifibration;
- $\mathcal{D}$ bicartesian closed,
  p preserves bi-cc structure
- fibres have small products $\bigwedge$

predicates have inverse and direct images

predicate conjunction

Use change-of-base:

binary logical relations

$$\begin{tikzcd} \mathbf{LogRel} \arrow[r] \arrow[d] \arrow[dr, phantom, "\lrcorner", very near start] & \mathcal{D} \arrow[d, "p"] \\ \mathcal{C} \times \mathcal{C} \arrow[r, "\times"'] & \mathcal{C} \end{tikzcd}$$

$\mathcal{M}$-cod bifibration [Hughes and Jacobs '03]

In a factorisation system, cod : $\mathcal{M} \to \mathcal{C}$ is a bifibration:

over a category with pullbacks of $\mathcal{M}$-monos

inverse images

$$\begin{tikzcd} f^* P \arrow[r] \arrow[d, tail, "f^* m"'] \arrow[dr, phantom, very near start, "\lrcorner"] & P \arrow[d, tail, "m"] \\ X \arrow[r, "f"'] & Y \end{tikzcd}$$

direct images

$$\begin{tikzcd} P \arrow[r, two heads] \arrow[d, tail, "m"'] \arrow[dr, phantom, "="] & f_* P \arrow[d, tail, "m'"] \\ X \arrow[r, "f"'] & Y \end{tikzcd}$$

Definition (Factorisation systems for logical relations)

A factorisation system of a bi-ccc such that:

- $\mathcal{M}$ contains only monomorphisms
- $\mathcal{M}$ closed under coproducts, $\mathcal{E}$ under products
- Fibres have small products

cod is faithful

p is bi-ccc

Monadic lifting

Σ-monad

fibration for logical relations

A Σ-lifting of T along p is a Σ-monad $\dot{T}$ on $\mathcal{D}$ above T:

$$
\begin{tikzcd}
X \arrow[r, "\dot\eta"] \arrow[d, mapsto] & \dot{T}X \arrow[d, mapsto] & \dot{T}(\dot{T}X) \arrow[l, "\dot\mu"'] \arrow[d, mapsto] \\
pX \arrow[r, "\eta"'] & T(pX) & T(T(pX)) \arrow[l, "\mu"]
\end{tikzcd}
\qquad
\begin{tikzcd}
\llbracket \dot{G} \rrbracket \arrow[r, "\dot{\llbracket op \rrbracket}"] \arrow[d, mapsto] & \dot{T}\llbracket \dot{G}' \rrbracket \arrow[d, mapsto] \\
\llbracket G \rrbracket \arrow[r, "\llbracket op \rrbracket"'] & T\llbracket G' \rrbracket
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathcal{D} \arrow[d, "p"] \\
\mathcal{C}
\end{tikzcd}
$$

for all $(op : G \to G') \in \Sigma$.

$$
\begin{tikzcd}
\mathbf{LogRel} \arrow[loop right, "\dot{T}"] \arrow[d, "p"] \\
\mathcal{C} \arrow[loop right, "T"]
\end{tikzcd}
$$

Free lifting

Key idea [Plotkin and Power'03]

$$G \to TG' \qquad \longleftrightarrow \qquad \alpha^{\mathsf{op}}_X : (G' \Rightarrow TX) \to (G \Rightarrow TX)$$

generic effect — algebraic operation

$Y \in \mathcal{R}\,X$ when

$$\begin{tikzcd} X \arrow[r, "\dot\eta"] \arrow[d, mapsto] & Y \arrow[d, mapsto] \\ pX \arrow[r, "\eta"'] & T(pX) \end{tikzcd}$$

and for all $(\mathsf{op} : G \to G') \in \Sigma$:

$$\begin{tikzcd} \llbracket \dot{G}' \rrbracket \Rightarrow Y \arrow[r, "\dot\alpha^{\mathsf{op}}_X"] \arrow[d, mapsto] & \llbracket \dot{G} \rrbracket \Rightarrow Y \arrow[d, mapsto] \\ \llbracket G' \rrbracket \Rightarrow pX \arrow[r, "\alpha^{\mathsf{op}}_X"'] & \llbracket G \rrbracket \Rightarrow T(pX) \end{tikzcd}$$

$$\dot{T}X := \bigwedge \mathcal{R}X$$

- $\dot{T}$ is a lifting of T to $\mathcal{D}$
- $\dot{T}$ is initial: if $\dot{T}'$ is a lifting then $\dot{T} \le \dot{T}'$

Theorem (Soundness and Completeness)

If:

- *Factorisation system for logical relations is well-powered*
- *Diagonals $\delta : X \rightarrowtail X \times X$ are monos*
- *$\mathcal{C}$ is locally presentable bi-ccc*

then

$$[\![\underbrace{\Gamma \vdash_{\varepsilon} M : G}_{\text{ground}}]\!] = [\![\Gamma \vdash_{\varepsilon} N : G]\!] \quad \Leftrightarrow \quad [\![\Gamma \vdash M : G]\!] = [\![\Gamma \vdash N : G]\!]$$

Contributions

Given an unrefined model, with:

- A suitable category $\mathcal{C}$ (factorisation system, etc.)
- Any Σ-monad on $\mathcal{C}$

Can construct a refined model

- Containing simpler monads T_ε
- With a completeness theorem:

$$[\![\Gamma \vdash_\varepsilon M : G]\!] = [\![\Gamma \vdash_\varepsilon N : G]\!] \quad \Leftrightarrow \quad [\![\Gamma \vdash M : G]\!] = [\![\Gamma \vdash N : G]\!]$$

ground