

A Propositional Refinement Type System for Algebraic Effects

Danel Ahman
(joint work with Gordon Plotkin)

PLInG Meeting, 27 January 2014

informatics

lfcs

Laboratory for Foundations
of Computer Science



THE UNIVERSITY *of* EDINBURGH

Single system - different effects - different specs



type-and-effect
systems
+ optimizations

$$\Gamma \vdash M : \sigma ! \varepsilon$$

input/output
+ sessions and protocols

$$\Gamma \vdash M : ?(! (1).end; end)$$

state
+ Hoare logic

$$\Gamma \vdash M : \{P\} \sigma \{Q\}$$



CBPV_{ref}



computational language (CBPV)
with algebraic effects

Single system - different effects - different specs



type-and-effect
systems
+ optimizations

$$\Gamma \vdash M : \sigma ! \varepsilon$$

input/output
+ sessions and protocols

$$\Gamma \vdash M : ?(! (1).end; end)$$

state
+ Hoare logic

$$\Gamma \vdash M : \{P\} \sigma \{Q\}$$



CBPV_{ref}

specifications on values
(even/odd numbers)



computational language (CBPV)
with algebraic effects

Single system - different effects - different specs



type-and-effect
systems
+ optimizations

$\Gamma \vdash M : \sigma ! \varepsilon$

input/output
+ sessions and protocols

$\Gamma \vdash M : ?(! (1).end; end)$

state
+ Hoare logic

$\Gamma \vdash M : \{P\} \sigma \{Q\}$

\sqsubseteq

\sqcap

\sqsupseteq

CBPV_{ref}



computational language (CBPV)
with algebraic effects

- Assume: an algebraic theory \mathcal{T}_{eff} of computational effects

- collection of operation symbols
- collection of equations

Plotkin & Power '02,'03

- Theory \mathcal{T}_{ND} of non-determinism

- one binary operation $\oplus : 2$
- equations $x \oplus x = x$,

$$x \oplus y = y \oplus x,$$

$$(x \oplus y) \oplus z = x \oplus (y \oplus z)$$

- Theory $\mathcal{T}_{\text{I/O}}$ of input/output (of bits, over a fixed channel)

- three operations receive : 2,

$$\text{send}_0 : 1,$$

$$\text{send}_1 : 1$$

- no equations

- Assume: an algebraic theory \mathcal{T}_{eff} of computational effects

- collection of operation symbols
- collection of equations

Plotkin & Power '02,'03

- Theory \mathcal{T}_S of global state (of bits)

- three operations lookup : 2,

update₀ : 1,

update₁ : 1

- equations

$$x = \text{lookup}(\text{update}_0(x), \text{update}_1(x))$$

$$\text{update}_0(\text{lookup}(x, y)) = \text{update}_0(x)$$

$$\text{update}_1(\text{lookup}(x, y)) = \text{update}_1(y)$$

$$\text{update}_i(\text{update}_j(x)) = \text{update}_j(x)$$

Single system - different effects - different specs



type-and-effect
systems
+ optimizations

$\Gamma \vdash M : \sigma ! \varepsilon$

input/output
+ sessions and protocols

$\Gamma \vdash M : ?(! (1).end; end)$

state
+ Hoare logic

$\Gamma \vdash M : \{P\} \sigma \{Q\}$



CBPV_{ref}



computational language (CBPV)
with algebraic effects

- Strict separation into **values** and **computations**

Levy '04

- **Types:**

- $A ::= 1 \mid 0 \mid A_1 \times A_2 \mid A_1 + A_2 \mid U \underline{B}$

- $\underline{B} ::= FA \mid \underline{B}_1 \times \underline{B}_2 \mid A \rightarrow \underline{B}$

- **Terms:**

- $V ::= \star \mid \langle V_1, V_2 \rangle \mid \text{inj}_i V \mid \text{thunk } M$

- $M ::= \text{return } V \mid M \text{ to } x : A. N \mid \text{op}_{\underline{B}}(M_1, \dots, M_n) \mid$
 $\lambda x : A. M \mid MV \mid \text{force } V \mid \langle M_1, M_2 \rangle \mid \text{fst } M \mid \text{snd } M \mid$
 $\text{match } V \text{ as } (x_1 : A_1, x_2 : A_2). M \mid \text{match } V \text{ as } \{ \} \mid$
 $\text{match } V \text{ as } \{ \text{inj}_1 (x_1 : A_1) \mapsto M_1, \text{inj}_2 (x_2 : A_2) \mapsto M_2 \}$

- **Equational theory:**

- standard $\beta\eta$ -equations +

algebraic equations $\Gamma \Vdash M \oplus N = N \oplus M : \underline{B}$

Single system - different effects - different specs



type-and-effect
systems
+ optimizations

$\Gamma \vdash M : \sigma ! \varepsilon$

input/output
+ sessions and protocols

$\Gamma \vdash M : ?(! (1).end; end)$

state
+ Hoare logic

$\Gamma \vdash M : \{P\} \sigma \{Q\}$



CBPV_{ref}



computational language (CBPV)
with algebraic effects

- To **accommodate logical properties** in your favorite type system
- **Intersection types** $\sigma_1 \wedge \sigma_2$, **union types** $\sigma_1 \vee \sigma_2$

Freeman & Pfenning '91, ...

 - practical propositional reasoning
 - ref. types not explicitly connected to underlying system
- **First-order logic** in types $\{x : \sigma \mid \varphi\}$

Denney '98, ...

 - reasoning in full first-order logic
 - ref. types explicitly connected to underlying system
 - $\vdash \sigma : \text{Ref}(A)$ by using indexed kinds
- CBPV_{ref} is in-between these two approaches

Refinement types in CBPV_{ref}

- Separated into **value** and **computation** refinement types

$$\sigma ::= 1 \mid 0 \mid \sigma_1 \times \sigma_2 \mid \sigma_1 + \sigma_2 \mid U^{\text{ref}} \underline{\tau} \quad \leftarrow \text{structural}$$

$$\sigma_1 \wedge \sigma_2 \mid \sigma_1 \vee \sigma_2 \mid \perp_A \quad \leftarrow \text{logical}$$

$$\underline{\tau} ::= F^{\text{ref}} \sigma \mid \underline{\tau}_1 \times \underline{\tau}_2 \mid \sigma \rightarrow \underline{\tau} \mid$$

$$\langle \text{op} \rangle_{\underline{B}}(\underline{\tau}_1, \dots, \underline{\tau}_n) \mid \quad \leftarrow \text{operation modalities}$$

$$X \mid \mu X : \text{Ref}(\underline{B}). \underline{\tau} \mid \underline{\tau}_1 \wedge \underline{\tau}_2 \mid \underline{\tau}_1 \vee \underline{\tau}_2 \mid \perp_{\underline{B}}$$

- Well-kinded** ref. types over CBPV types, e.g.:

$$\frac{\vdash \sigma : \text{Ref}(A) \quad \Delta \vdash \underline{\tau} : \text{Ref}(\underline{B})}{\Delta \vdash \sigma \rightarrow \underline{\tau} : \text{Ref}(A \rightarrow \underline{B})}$$

$$\frac{\Delta \vdash \sigma_1 : \text{Ref}(A) \quad \Delta \vdash \sigma_2 : \text{Ref}(A)}{\Delta \vdash \sigma_1 \wedge \sigma_2 : \text{Ref}(A)}$$

- Prop.:** Unique kinding

- Main tool: **subtyping**
- Two subtyping relations
 - $\Delta \Vdash^r \sigma_1 \sqsubseteq_A \sigma_2$
 - $\Delta \Vdash^c \tau_1 \sqsubseteq_{\underline{B}} \tau_2$
- Again, **structural**

$$\frac{\Vdash^r \sigma_2 \sqsubseteq_A \sigma_1 \quad \Delta \Vdash^c \tau_1 \sqsubseteq_{\underline{B}} \tau_2}{\Delta \Vdash^c \sigma_1 \rightarrow \tau_1 \sqsubseteq_{A \rightarrow \underline{B}} \sigma_2 \rightarrow \tau_2} \qquad \frac{\Delta \Vdash^r \sigma_1 \sqsubseteq_A \sigma_2}{\Delta \Vdash^c F^{\text{ref}} \sigma_1 \sqsubseteq_{FA} F^{\text{ref}} \sigma_2}$$

$$\Delta \Vdash^c \langle \text{op} \rangle_{A \rightarrow \underline{B}} (\sigma \rightarrow \tau_1, \dots, \sigma \rightarrow \tau_n) = \sigma \rightarrow \langle \text{op} \rangle_{\underline{B}} (\tau_1, \dots, \tau_n)$$

- and **logical** rules

$$\frac{\Delta \Vdash^c \tau \sqsubseteq \tau'[\mu X : \text{Ref}(\underline{B}).\tau' / X]}{\Delta \Vdash^c \tau \sqsubseteq \mu X : \text{Ref}(\underline{B}).\tau'}$$

$$\frac{\Delta \Vdash^c \tau[\tau' / X] \sqsubseteq \tau'}{\Delta \Vdash^c \mu X : \text{Ref}(\underline{B}).\tau \sqsubseteq \tau'}$$

- Separated into **value** and **computation** terms

$$V ::= x \mid \star \mid \langle V_1, V_2 \rangle \mid \text{inj}_i V \mid \text{thunk } M$$

$$\begin{aligned} M, N ::= & \text{return } V \mid M \text{ to } x : \sigma. N \mid \text{op}_{\underline{B}}(M_1, \dots, M_n) \mid \\ & \lambda x : \sigma. M \mid MV \mid \text{force } V \mid \langle M_1, M_2 \rangle \mid \text{fst } M \mid \text{snd } M \\ & \text{match } V \text{ as } (x_1 : \sigma_1, x_2 : \sigma_2). M \mid \text{match } V \text{ as } \{ \} \mid \\ & \text{match } V \text{ as } \{ \text{inj}_1 (x_1 : \sigma_1) \mapsto M_1, \text{inj}_2 (x_2 : \sigma_2) \mapsto M_2 \} \mid \end{aligned}$$

- Typing rules** mostly standard from CBPV, e.g.:

$$\frac{\Gamma \Vdash V : \sigma}{\Gamma \Vdash \text{return } V : F^{\text{ref}} \sigma} \qquad \frac{\Gamma, x : \sigma \Vdash M : \underline{\tau}}{\Gamma \Vdash \lambda x : \sigma. M : \sigma \rightarrow \underline{\tau}}$$

- except for ...

- First exception: algebraic operations

$$\frac{\Gamma \Vdash^r M_1 : \underline{\tau}_1 \quad \dots \quad \Gamma \Vdash^r M_n : \underline{\tau}_n}{\Gamma \Vdash^r \text{op}_{\underline{B}}(M_1, \dots, M_n) : \langle \text{op} \rangle_{\underline{B}}(\underline{\tau}_1, \dots, \underline{\tau}_n)} \quad (\text{op} : n)$$

- Second exception: **sequential composition**

$$\frac{\Gamma \Vdash M : C[F^{\text{ref}}\sigma] \quad \Gamma, x : \sigma \Vdash N : \underline{\tau}}{\Gamma \Vdash M \text{ to } x : \sigma. N : C[\underline{\tau}]}$$

- Uses **computation refinement contexts** C

$$C ::= [] \mid \langle \text{op} \rangle (C_1, \dots, C_n) \mid X \mid \mu X. C \mid C_1 \wedge C_2 \mid C_1 \vee C_2 \mid \perp$$

- **Hole filling** $C[\underline{\tau}]$ defined by struct. recursion

- Prop.: For any $\vdash C$ we have:

$$\Delta \Vdash^r C[\underline{\tau}_1 \times \underline{\tau}_2] \sqsubseteq_{B_1 \times B_2} C[\underline{\tau}_1] \times C[\underline{\tau}_2]$$

$$\Delta \Vdash^r C[\sigma \rightarrow \underline{\tau}] \sqsubseteq_{A \rightarrow B} \sigma \rightarrow C[\underline{\tau}]$$

- Third exception: [subtyping](#)

$$\frac{\Gamma \Vdash V : \sigma_1 \quad \Vdash \sigma_1 \sqsubseteq \sigma_2}{\Gamma \Vdash V : \sigma_2} \quad \frac{\Gamma \Vdash M : \tau_1 \quad \Vdash \tau_1 \sqsubseteq \tau_2}{\Gamma \Vdash M : \tau_2}$$

- Fourth exception: [interaction](#) between local and global, e.g.:

$$\frac{\Gamma \Vdash M : \tau}{\Gamma_{\perp} \Vdash M : \perp_{|\tau|}} \quad \frac{\Gamma \Vdash M : \tau_1 \quad \Gamma \Vdash M : \tau_2}{\Gamma \Vdash M : \tau_1 \wedge \tau_2}$$

$$\frac{\Gamma, x : \sigma_1 \Vdash M : \tau \quad \Gamma, x : \sigma_2 \Vdash M : \tau \quad \Vdash \sigma \sqsubseteq \sigma_1 \vee \sigma_2}{\Gamma, x : \sigma \Vdash M : \tau}$$

- and dual rules for values

- Interaction rules allow us to prove interesting derivations, e.g.:

$$\begin{array}{c} \Gamma \Vdash V : \perp_1 + 1 \quad \Gamma, x_1 : 1 \Vdash \text{update}_1(\text{return } \star) : \langle \text{update}_1 \rangle_{\underline{B}}(F^{\text{ref}} 1) \\ \Gamma, x_2 : 1 \Vdash \text{return } \star : F^{\text{ref}} 1 \end{array}$$

$$\Gamma \Vdash \text{match } V \text{ as } \{\text{inj}_1(x_1) \mapsto \text{update}_1(\text{return } \star), \text{inj}_2(x_2) \mapsto \text{return } \star\} : F^{\text{ref}} 1$$

- For equational reasoning and program optimizations
- Collection of equations $\Gamma \Vdash V_1 = V_2 : \sigma$ and $\Gamma \Vdash M_1 = M_2 : \tau$
- Divided into:
 - basic (congruence relation + subtyping)
 - structural ($\beta\eta$ -equations from CBPV)
 - algebraic, e.g.:

$$\Gamma \Vdash M \oplus N = N \oplus M : \langle \oplus \rangle_{\underline{B}}(\tau_N, \tau_M)$$

- refinement-typed, e.g.:

$$\frac{\Gamma \Vdash M : F^{\text{ref}}\sigma \quad \Gamma \Vdash N : \tau}{\Gamma \Vdash M \text{ to } x : \sigma. N = N : \tau}$$

$$\frac{\Gamma \Vdash V_1 : \sigma \quad \Gamma \Vdash V_2 : \sigma}{\Gamma_{\perp} \Vdash V_1 = V_2 : \perp_{|\sigma|}}$$

and rest of the “interaction” equations

Single system - different effects - different specs



type-and-effect
systems
+ optimizations

$\Gamma \vdash M : \sigma ! \varepsilon$



input/output
+ sessions and protocols

$\Gamma \vdash M : ?(! (1).end; end)$



state
+ Hoare logic

$\Gamma \vdash M : \{P\} \sigma \{Q\}$



CBPV_{ref}



computational language (CBPV)
with algebraic effects

- $\Gamma \vdash loc_2 := !(loc_1) ; loc_1 := 0 : () ! \{\text{lookup}, \text{update}\}$
 - typing rules $\Gamma \vdash M : \sigma ! \varepsilon$ with extra effect annotations

- **Algebraic idea**: let ε be an algebraic signature Kammar & Plotkin '12

- We use a **collection of equiv. classes of trees** built from ε

- In CBPV_{ref} , we define $\sigma ! \varepsilon \stackrel{\text{def}}{=} C_\varepsilon[F^{\text{ref}} \sigma]$ where

$$C_\varepsilon \stackrel{\text{def}}{=} \mu X. ([] \vee \langle \text{op}_1 \rangle_{FA}(X, \dots, X) \vee \dots \vee \langle \text{op}_m \rangle_{FA}(X, \dots, X))$$

$$\text{for } \varepsilon = \{\text{op}_1 : n_1, \dots, \text{op}_m : n_m\}$$

- Another example: $(\text{update}_0 \text{ not observable and thus not in } \varepsilon)$

$$\Gamma \Vdash \text{lookup}(\text{update}_0(\text{return } \star), \text{return } \star) : 1 ! \{\text{lookup}\}$$

- Can translate well-(ε -)kinded Kammar & Plotkin types:

$$\blacksquare (F_\varepsilon A)_\varepsilon^\circ \stackrel{\text{def}}{=} C_\varepsilon[F^{\text{ref}} A^\circ] , (A \rightarrow \underline{B})_\varepsilon^\circ \stackrel{\text{def}}{=} A^\circ \rightarrow \underline{B}_\varepsilon^\circ \quad (, \text{ and } \times)$$

with translated types having expected properties, e.g.:

$$\blacksquare \Vdash_{\mathcal{C}} C_\varepsilon[\underline{B}_\varepsilon^\circ] \sqsubseteq \underline{B}_\varepsilon^\circ$$

$$\blacksquare \Vdash_{\mathcal{C}} C_{\varepsilon_1}[F^{\text{ref}} A^\circ] \sqsubseteq C_{\varepsilon_2}[F^{\text{ref}} A^\circ] \quad \text{when } \varepsilon_1 \subseteq \varepsilon_2$$

- Kammar & Plotkin's abstract optimizations, e.g. **Copy**

$$\Gamma \Vdash_{\mathcal{C}} M : C_\varepsilon[F^{\text{ref}} \sigma]$$

$$t(t(x_{11}, \dots, x_{1n}), \dots, t(x_{n1}, \dots, x_{nn})) = t(x_{11}, \dots, x_{nn})$$

holds in \mathcal{T}_{eff} for all terms t built from ε

$$\Gamma \Vdash_{\mathcal{C}} M \text{ to } x : \sigma. M \text{ to } y : \sigma. \text{return } \langle x, y \rangle = M \text{ to } x. \text{return } \langle x, x \rangle : C_\varepsilon[F^{\text{ref}}(\sigma \times \sigma)]$$

Single system - different effects - different specs



type-and-effect
systems
+ optimizations

$\Gamma \vdash M : \sigma ! \varepsilon$

input/output
+ sessions and protocols

$\Gamma \vdash M : ?(! (1).end; end)$

state
+ Hoare logic

$\Gamma \vdash M : \{P\} \sigma \{Q\}$

\sqsubseteq

\sqcap

\sqsupseteq

CBPV_{ref}



computational language (CBPV)
with algebraic effects

- Want to specify how a process should use I/O channel(s)
- Interested in following specs., *inspired by session types*

$$S ::= !(0).S \mid !(1).S \mid !(0 \vee 1).S \mid ?(S_1, S_2) \mid end$$

- Easy to define *using operation modalities* and *C's*

$$C_{!(0).S} \stackrel{\text{def}}{=} \langle \text{send}_0 \rangle (C_S)$$

$$C_{!(1).S} \stackrel{\text{def}}{=} \langle \text{send}_1 \rangle (C_S)$$

$$C_{!(0 \vee 1).S} \stackrel{\text{def}}{=} \langle \text{send}_0 \rangle (C_S) \vee \langle \text{send}_1 \rangle (C_S)$$

$$C_{?(S_1, S_2)} \stackrel{\text{def}}{=} \langle \text{receive} \rangle (C_{S_1}, C_{S_2})$$

$$end \stackrel{\text{def}}{=} []$$

- Any use for fixed point refinements?
- Protocols for **correctly using files** (talking to a server, etc.)
- Using a file correctly **once**:

$$C_{\text{files}} ::= \langle \text{open} \rangle (\mu X. (\langle \text{close} \rangle ([]) \vee \langle \text{write}_i \rangle (X) \vee \langle \text{read} \rangle (X, X)))$$

- Using a file correctly **repetitively**:

$$C_{\text{rep-files}} ::= \mu Y. ([] \vee C_{\text{files}}[Y])$$

Single system - different effects - different specs



type-and-effect
systems
+ optimizations

$\Gamma \vdash M : \sigma ! \varepsilon$

input/output
+ sessions and protocols

$\Gamma \vdash M : ?(! (1).end; end)$

state
+ Hoare logic

$\Gamma \vdash M : \{P\} \sigma \{Q\}$



CBPV_{ref}



computational language (CBPV)
with algebraic effects

- Would like to annotate terms with **Hoare triples** $\{P\} \sigma \{Q\}$
 - Hoare Logic in some ref. ty. sys. supporting only state effects:
 - using the **refined state monad**
 - using the **Dijkstra monad**
- Borgström et al. '11
- Swamy et al. '13
- In this (propositionally clumsy) example, we take $P, Q \subseteq \{0, 1\}$
 - Define $\{P\} \sigma \{Q\} \stackrel{\text{def}}{=} \{P\} [F^{\text{ref}} \sigma] \{Q\}$ by case analysis on P

$$\begin{aligned}\{\emptyset\} [] \{Q\} &\stackrel{\text{def}}{=} \langle \text{lookup} \rangle (\bigvee_i \langle \text{update}_i \rangle ([], \bigvee_i \langle \text{update}_i \rangle ([])) \\ \{\{0\}\} [] \{Q\} &\stackrel{\text{def}}{=} \langle \text{lookup} \rangle (\bigvee_q \langle \text{update}_q \rangle ([], \bigvee_i \langle \text{update}_i \rangle ([])) \\ \{\{1\}\} [] \{Q\} &\stackrel{\text{def}}{=} \langle \text{lookup} \rangle (\bigvee_i \langle \text{update}_i \rangle ([], \bigvee_q \langle \text{update}_q \rangle ([])) \\ \{\{0, 1\}\} [] \{Q\} &\stackrel{\text{def}}{=} \langle \text{lookup} \rangle (\bigvee_q \langle \text{update}_q \rangle ([], \bigvee_q \langle \text{update}_q \rangle ([]))\end{aligned}$$

where $i \in \{0, 1\}$ and $q \in Q$

- Prop.: With this def. of $\{P\} \sigma \{Q\}$ following rules are admissible

$$\frac{\Gamma \models M : \{P \cap \{0\}\} \sigma \{Q\} \quad \Gamma \models N : \{P \cap \{1\}\} \sigma \{Q\}}{\Gamma \models \text{lookup}(M, N) : \{P\} \sigma \{Q\}}$$

$$\frac{\Gamma \models M : \{\{0\}\} \sigma \{Q\}}{\Gamma \models \text{update}_0(M) : \{P\} \sigma \{Q\}}$$

$$\frac{\Gamma \models M : \{\{1\}\} \sigma \{Q\}}{\Gamma \models \text{update}_1(M) : \{P\} \sigma \{Q\}}$$

$$\frac{\Gamma \models M : \{P\} \sigma_1 \{Q\} \quad \Gamma, x : \sigma_1 \models N : \{Q\} \sigma_2 \{R\}}{\Gamma \models M \text{ to } x : \sigma_1. N : \{P\} \sigma_2 \{R\}}$$

$$\frac{P \subseteq P' \quad \Gamma \models M : \{P'\} \sigma \{Q'\} \quad Q' \subseteq Q}{\Gamma \models M : \{P\} \sigma \{Q\}}$$

Single system - different effects - different specs



type-and-effect
systems
+ optimizations

$\Gamma \vdash M : \sigma ! \varepsilon$

input/output
+ sessions and protocols

$\Gamma \vdash M : ?(! (1).end; end)$

state
+ Hoare logic

$\Gamma \vdash M : \{P\} \sigma \{Q\}$

\sqsubseteq

\sqsubseteq

\sqsubseteq

CBPV_{ref}



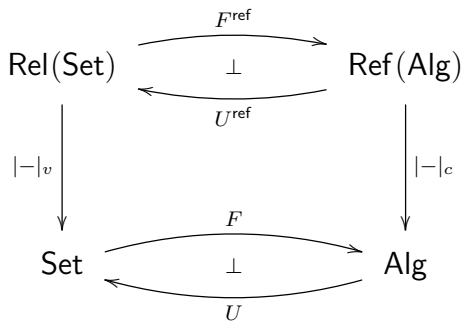
two-level semantics

computational language (CBPV)
with algebraic effects

Two-level CBPV_{ref} semantics



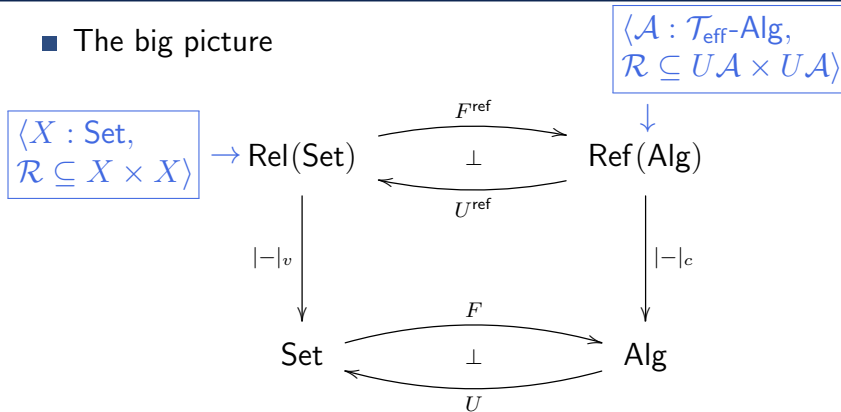
■ The big picture



where Alg is the category of \mathcal{T}_{eff} -algebras in Set

Two-level CBPV_{ref} semantics

■ The big picture

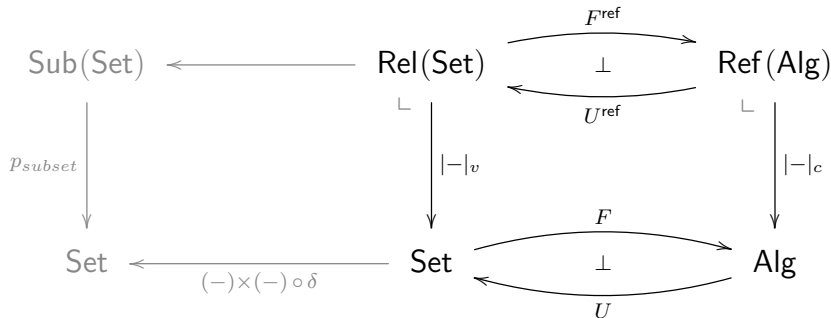


where Alg is the category of \mathcal{T}_{eff} -algebras in Set

Two-level CBPV_{ref} semantics



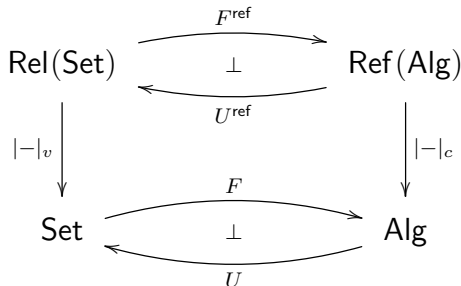
- How one constructs this quadruple:



where

- $U^{\text{ref}} \stackrel{\text{def}}{=} (|-|_v)^*(U)$
- $F^{\text{ref}}\sigma \stackrel{\text{def}}{=} \langle F|\sigma|_v, (\eta|_{\sigma|_v})!(\sigma) \rangle$

■ Back to the big picture



■ Refinement types *interpreted as functors*

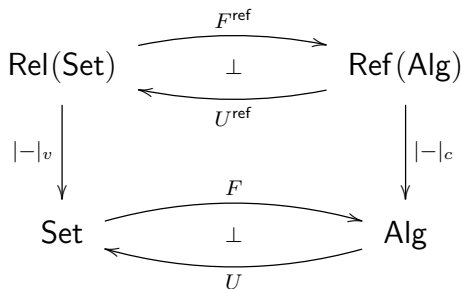
$$\llbracket \vec{X} : \text{Ref}(\vec{B}) \vdash \sigma : \text{Ref}(A) \rrbracket : \text{Ref}_{\llbracket \underline{B}_1 \rrbracket}(\text{Alg}) \times \dots \times \text{Ref}_{\llbracket \underline{B}_n \rrbracket}(\text{Alg}) \rightarrow \text{Rel}_{\llbracket A \rrbracket}(\text{Set})$$

$$\llbracket \vec{X} : \text{Ref}(\vec{B}) \vdash \tau : \text{Ref}(B) \rrbracket : \text{Ref}_{\llbracket \underline{B}_1 \rrbracket}(\text{Alg}) \times \dots \times \text{Ref}_{\llbracket \underline{B}_n \rrbracket}(\text{Alg}) \rightarrow \text{Ref}_{\llbracket B \rrbracket}(\text{Alg})$$

Two-level CBPV_{ref} semantics



- More on the big picture



- Computation refinement contexts interpreted as fam. of functors

$$\llbracket \vec{X} \vdash C \rrbracket_{\underline{B}} : \text{Ref}_{\llbracket \underline{B} \rrbracket}(\text{Alg}) \times \dots \times \underbrace{\text{Ref}_{\llbracket \underline{B} \rrbracket}(\text{Alg}) \times \dots \times \text{Ref}_{\llbracket \underline{B} \rrbracket}(\text{Alg})}_{\llbracket \rrbracket} \rightarrow \text{Ref}_{\llbracket \underline{B} \rrbracket}(\text{Alg})$$

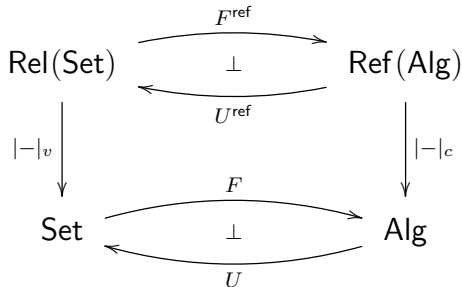
- Importantly: closed C 's still have functorial flavor:

$$\llbracket \vdash \tau_1 \rrbracket \rightarrow \llbracket \vdash \tau_2 \rrbracket \implies \llbracket \vdash C \rrbracket_{\underline{B}_1}(\llbracket \vdash \tau_1 \rrbracket) \rightarrow \llbracket \vdash C \rrbracket_{\underline{B}_2}(\llbracket \vdash \tau_2 \rrbracket)$$

Two-level CBPV_{ref} semantics



- Even more on the big picture



- Terms interpreted as morphisms in $\text{Rel}(\text{Set})$

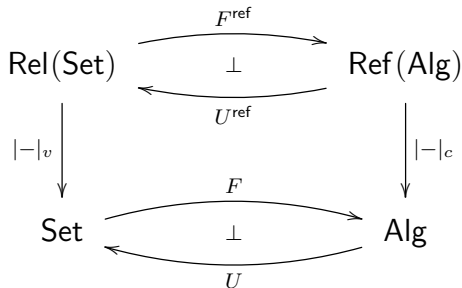
$$\llbracket \Gamma \stackrel{\text{r}}{\Vdash} V : \sigma \rrbracket : \llbracket \Gamma \rrbracket \longrightarrow \llbracket \sigma \rrbracket$$

$$\llbracket \Gamma \stackrel{\text{r}}{\Vdash} M : \tau \rrbracket : \llbracket \Gamma \rrbracket \longrightarrow U^{\text{ref}} \llbracket \tau \rrbracket$$

Two-level CBPV_{ref} semantics



- Even more on the big picture



- Prop.: The interpretation is **sound**

$$\Gamma \Vdash V_1 = V_2 : \sigma \implies \llbracket V_1 \rrbracket = \llbracket V_2 \rrbracket \quad \text{and} \quad \Gamma \Vdash M_1 = M_2 : \tau \implies \llbracket M_1 \rrbracket = \llbracket M_2 \rrbracket$$

- Prop.: The interpretation is **coherent**

interpretations of different derivations of a CBPV_{ref} term all agree

- A proposal for a unifying categorical semantics of a **monadic effect-annotated language**

Wadler & Thiemann '03

- $A, B ::= 1 \mid 0 \mid A \times B \mid A \rightarrow B \mid T_{\varepsilon}A$

- Instead of taking a **single monad** $T : \mathcal{V} \rightarrow \mathcal{V}$

- Take a **lax mon. functor** $T : \mathbb{E} \rightarrow [\mathcal{V}, \mathcal{V}]$

- \mathbb{E} a preordered monoid of effects (a mon. category)

- $\eta : \text{Id} \rightarrow T_1$

- $\mu_{\varepsilon_1, \varepsilon_2} : T_{\varepsilon_1} \circ T_{\varepsilon_2} \rightarrow T_{\varepsilon_1 \circ \varepsilon_2}$

- Parallels with our work:

- can interpret closed C 's as unary functors

$$\llbracket C \rrbracket : \text{Ref}(\text{Alg}) \rightarrow \text{Ref}(\text{Alg})$$

- closed C 's form a preordered monoid \mathbb{C}

- so could also interpret as $\llbracket - \rrbracket : \mathbb{C} \rightarrow [\text{Ref}(\text{Alg}), \text{Ref}(\text{Alg})]$

- Algebraic work for accommodating effects in ref. ty. systems
- Examples: Effect-and-type annotations, protocols, Hoare Logic

Future work:

- Operations with parameters and binding
- Local effects
- Effect handlers
- Optimizations making use of fine-grained refinements
- Combining theories vs. combining refinements
- More general categorical account of CBPV_{ref}