Abstract Effects and Proof-Relevant Logical Relations

Nick Benton, Martin Hofmann,
Andrew Kennedy, Vivek Nigam

Current draft at : `http://research.microsoft.com/~nick/setoids.pdf`

Effect Systems

- Static analyses refining conventional types with upper bounds on possible side effects of expressions
- Gifford, Jouvelot, Lucassen, Sheldon. FX'87
- Dozens (hundreds?) since, including static memory management in ML-Kit and monadic effects in MLj and SML.NET for optimization

$$\frac{}{\Theta \vdash n : \mathtt{int}} \qquad \frac{}{\Theta \vdash b : \mathtt{bool}} \qquad \frac{}{\Theta \vdash () : \mathtt{unit}} \qquad \frac{}{\Theta, x : X \vdash x : X}$$

$$\frac{\Theta \vdash V_1 : \mathtt{int} \quad \Theta \vdash V_2 : \mathtt{int}}{\Theta \vdash V_1 + V_2 : \mathtt{int}} \qquad \frac{\Theta \vdash V_1 : \mathtt{int} \quad \Theta \vdash V_2 : \mathtt{int}}{\Theta \vdash V_1 > V_2 : \mathtt{bool}}$$

$$\frac{\Theta \vdash V_1 : X \quad \Theta \vdash V_2 : Y}{\Theta \vdash (V_1, V_2) : X \times Y} \qquad \frac{\Theta \vdash V : X_1 \times X_2}{\Theta \vdash \pi_i\, V : X_i} \qquad \frac{\Theta, x : X \vdash M : T_\varepsilon Y}{\Theta \vdash \lambda x : U(X).M : X \to T_\varepsilon Y}$$

$$\frac{\Theta \vdash V_1 : X \to T_\varepsilon Y \quad \Theta \vdash V_2 : X}{\Theta \vdash V_1\, V_2 : T_\varepsilon Y} \qquad \frac{\Theta \vdash V : X}{\Theta \vdash \mathtt{val}\, V : T_\emptyset X}$$

$$\frac{\Theta \vdash M : T_\varepsilon X \quad \Theta, x : X \vdash N : T_{\varepsilon'} Y}{\Theta \vdash \mathtt{let}\, x \Leftarrow M\, \mathtt{in}\, N : T_{\varepsilon \cup \varepsilon'} Y} \qquad \frac{\Theta \vdash V : \mathtt{bool} \quad \Theta \vdash M : T_\varepsilon X \quad \Theta \vdash N : T_\varepsilon X}{\Theta \vdash \mathtt{if}\ V\ \mathtt{then}\ M\ \mathtt{else}\ N : T_\varepsilon X}$$

$$\frac{}{\Theta \vdash \mathtt{read}(\ell) : T_{\{\mathtt{r}_\ell\}}(\mathtt{int})} \qquad \frac{\Theta \vdash V : \mathtt{int}}{\Theta \vdash \mathtt{write}(\ell, V) : T_{\{\mathtt{w}_\ell\}}(\mathtt{unit})}$$

$$\frac{\Theta \vdash V : X \quad X \leq X'}{\Theta \vdash V : X'} \qquad \frac{\Theta \vdash M : T_\varepsilon X \quad T_\varepsilon X \leq T_{\varepsilon'} X'}{\Theta \vdash M : T_{\varepsilon'} X'}$$

Semantics?

- Want to justify effect-dependent transformations as contextual equivalences, not just soundness of analysis
- Don’t want something tied to syntactic rules of particular inference system – should be in terms of the observable behaviour of unrefined terms
- Amongst other things that lets us soundly combine refinements obtained by different analyses, including manual proof

Relations over denotational model

$$
\begin{array}{rcl}
[\![X]\!] & \subseteq & [\![U(X)]\!] \times [\![U(X)]\!] \\
[\![\mathtt{int}]\!] = \Delta_{\mathbb{Z}} \qquad [\![\mathtt{bool}]\!] = \Delta_{\mathbb{B}} \qquad [\![\mathtt{unit}]\!] = \Delta_1 & & \\
[\![X \times Y]\!] & = & [\![X]\!] \times [\![Y]\!] \\
[\![X \to T_\varepsilon Y]\!] & = & [\![X]\!] \to [\![T_\varepsilon Y]\!] \\
[\![T_\varepsilon X]\!] & = & \bigcap_{R \in \mathcal{R}_\varepsilon} R \to R \times [\![X]\!]
\end{array}
$$

where $\mathcal{R}_\varepsilon \subseteq \mathbb{P}(S \times S)$ is given by $\mathcal{R}_\varepsilon = \bigcap_{e \in \varepsilon} \mathcal{R}_e$ and for atomic effects e, $\mathcal{R}_e \subseteq \mathbb{P}(S \times S)$ is given by

$$
\begin{array}{rcl}
\mathcal{R}_{\mathtt{r}_\ell} & = & \{R \mid (s, s') \in R \implies s\,\ell = s'\,\ell\} \\
\mathcal{R}_{\mathtt{w}_\ell} & = & \{R \mid (s, s') \in R \implies \forall n \in \mathbb{Z}.\, (s[\ell \mapsto n], s'[\ell \mapsto n]) \in R\}
\end{array}
$$

Dead Computation:

$$\frac{\Theta \vdash M : T_\varepsilon X \quad \Theta \vdash N : T_{\varepsilon'} Y}{\Theta \vdash \mathtt{let}\, x \Leftarrow M\, \mathtt{in}\, N = N : T_{\varepsilon'} Y}\, x \notin \Theta, \mathrm{wrs}(\varepsilon) = \emptyset$$

Duplicated Computation:

$$\frac{\Theta \vdash M : T_\varepsilon X \quad \Theta, x : X, y : X \vdash N : T_{\varepsilon'} Y}{\Theta \vdash \begin{array}{rl} & \mathtt{let}\, x \Leftarrow M\, \mathtt{in}\, \mathtt{let}\, y \Leftarrow M\, \mathtt{in}\, N \\ = & \mathtt{let}\, x \Leftarrow M\, \mathtt{in}\, N[x/y] \end{array} : T_{\varepsilon \cup \varepsilon'} Y}\, \mathrm{rds}(\varepsilon) \cap \mathrm{wrs}(\varepsilon) = \emptyset$$

Commuting Computations:

$$\frac{\Theta \vdash M_1 : T_{\varepsilon_1} X_1 \quad \Theta \vdash M_2 : T_{\varepsilon_2} X_2 \quad \Theta, x_1 : X_1, x_2 : X_2 \vdash N : T_{\varepsilon'} Y}{\Theta \vdash \begin{array}{rl} & \mathtt{let}\, x_1 \Leftarrow M_1\, \mathtt{in}\, \mathtt{let}\, x_2 \Leftarrow M_2\, \mathtt{in}\, N \\ = & \mathtt{let}\, x_2 \Leftarrow M_2\, \mathtt{in}\, \mathtt{let}\, x_1 \Leftarrow M_1\, \mathtt{in}\, N \end{array} : T_{\varepsilon_1 \cup \varepsilon_2 \cup \varepsilon'} Y}\, \begin{array}{l} \mathrm{rds}(\varepsilon_1) \cap \mathrm{wrs}(\varepsilon_2) = \emptyset \\ \mathrm{wrs}(\varepsilon_1) \cap \mathrm{rds}(\varepsilon_2) = \emptyset \\ \mathrm{wrs}(\varepsilon_1) \cap \mathrm{wrs}(\varepsilon_2) = \emptyset \end{array}$$

Pure Lambda Hoist:

$$\frac{\Theta \vdash M : T_{\{\}} Z \quad \Theta, x : X, y : Z \vdash N : T_\varepsilon Y}{\Theta \vdash \begin{array}{rl} & \mathtt{val}\, (\lambda x : U(X).\mathtt{let}\, y \Leftarrow M\, \mathtt{in}\, N) \\ = & \mathtt{let}\, y \Leftarrow M\, \mathtt{in}\, \mathtt{val}\, (\lambda x : U(X).N) \end{array} : T_{\{\}}(X \to T_\varepsilon Y)}$$

A language with dynamic allocation

- Axiomatic (abstract) treatment of state equipped with dom, lookup, update and new
- Set of *regions* Regs, refined types include ref_r
- $\epsilon \subseteq \{rd_r, wr_r, al_r \mid r \in \mathrm{Regs}\}$

$$\frac{\Gamma(x) = \mathtt{int}}{\Gamma \vdash \mathtt{ref}(x) : \mathtt{ref}_\mathsf{r} : \{al_\mathsf{r}\}} \qquad \frac{\Gamma(x) = \mathtt{ref}_\mathsf{r} \quad \Gamma(y) = \mathtt{int}}{\Gamma \vdash x{:=}y : \mathtt{unit}, \{wr_\mathsf{r}\}}$$

$$\frac{\Gamma \vdash e : A, \varepsilon \quad \mathsf{r} \text{ does not occur in } \Gamma \text{ or } A}{\Gamma \vdash e : A, \varepsilon \setminus \{wr_\mathsf{r}, rd_\mathsf{r}, al_\mathsf{r}\}}$$

Parametric Logical Relation

- A parameter φ assigns every $r \in \text{Regs} \cup \{\tau\}$ a finite partial bijection on $\mathbb{L}$ (all disjoint)
- State relation on $L, L' \subseteq \mathbb{L}$ is $R \subseteq S \times S$ st. sRs', $s \sim_L s_1$, $s' \sim_{\mathcal{L}'} s_1' \Rightarrow s_1 R s_1'$
- If R state relation on $\text{dom}(\varphi)$, $\text{dom}'(\varphi)$ then
  - R respects $\{rd_r\}$ at φ if $sRs' \Rightarrow s.l = s'.l' \; \forall (l,l') \in \varphi(r)$
  - R respects $\{wr_r\}$ at φ if sRs', $(l,l') \in \boldsymbol{\varphi}(r)$, $v \in \mathbb{Z} \Rightarrow s[l \mapsto v] R s[l' \mapsto v]$
  - R respects $\{al_r\}$ always
- $\mathcal{R}_{\boldsymbol{\epsilon}}(\varphi) = \{R \in \text{StRel}(\text{dom}(\varphi), \text{dom}'(\varphi)) \text{ st. } \forall e \in \boldsymbol{\epsilon}, \text{ R resp. e at } \varphi\}$
- $[\![A]\!]_{\varphi}$ will be a QPER on $[\![|A|]\!]$
  - Relation R such that $R; R^{-1}; R = R$

$$
\begin{aligned}
[\![A]\!]_\varphi &\equiv \{(v,v) \mid v \in [\![|A|]\!]\} \text{ when } A \in \{\mathtt{int}, \mathtt{bool}, \mathtt{unit}\} \\
[\![\mathtt{ref}_{\mathsf{r}}]\!]_\varphi &\equiv \varphi(\mathsf{r}) \\
[\![A \times B]\!]_\varphi &\equiv [\![A]\!]_\varphi \times [\![B]\!]_\varphi \\
[\![A \xrightarrow{\varepsilon} B]\!]_\varphi &\equiv \{(f,f') \mid \forall\ \varphi' \geq \varphi . \forall (x,x') \in [\![A]\!]_{\varphi'} . \\
&\qquad\qquad (f(x), f'(x')) \in (T_\varepsilon [\![B]\!])_{\varphi'}\} \\
(T_\varepsilon Q)_\varphi &\equiv QPER(\{(f,f') \mid s, s' \models \varphi \Rightarrow \\
&\qquad \forall\ R \in \mathcal{R}_\varepsilon(\varphi) . s\ R\ s' \Rightarrow s_1\ R\ s_1'\ \wedge \\
&\qquad \exists \psi . (\psi(\mathsf{r}) \neq \emptyset \Rightarrow \mathsf{r} \in \mathrm{als}(\varepsilon))\ \wedge s_1, s_1' \models \varphi \otimes \psi \wedge \\
&\qquad s_1 \sim_\psi s_1' \wedge (v,v') \in Q_{\varphi \otimes \psi} \\
&\qquad \text{where } (s_1, v) = f\ s \text{ and } (s_1', v') = f'\ s'\})
\end{aligned}
$$

Skip this slide

- Monotonicity: $\varphi' \geq \varphi \Rightarrow [\![A]\!]_{\varphi'} \supseteq [\![A]\!]_{\varphi}$
- Masking: If r not in A, $[\![A]\!]_{\varphi} = [\![A]\!]_{\varphi - r}$ where φ – r moves φ(r) into the silent region τ
- Fundamental theorem
- Semantic equality, quantifying over parameters, is PER and yields same equations except duplicated computations requires no allocations as well as disjoint reads and writes

Abstract Effects and Locations

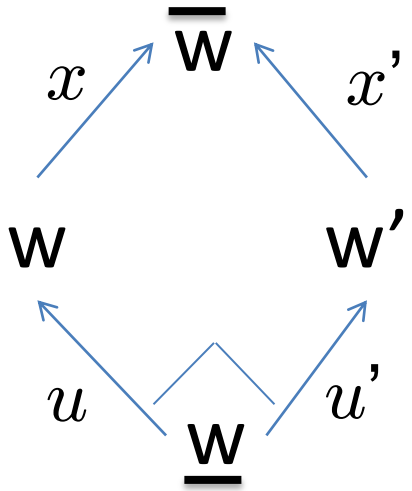
- Effectfulness often partially abstract
  - Masking deals with some completely non-observable uses of private state
- Consider a set ADT represented using balanced trees
  - Should be able to assign read-only effect to lookup operation even though internal rebalancing involves mutation
- Similarly, view of regions as disjoint sets heap locations is over-concrete
  - From the outside should be able to assign distinct regions to structures whose implementations physically overlap
  - E.g. two references encoded in one

Setoids

- Predomain $|A|$ & for $x,y \in |A|$, set $A(x,y)$ of 'proofs' that $x \sim y$
- $\{(x,y,p) \mid p : x \sim y\}$ a predom with cts projections
- $r_A : \Pi\, x \in |A|.A(x,x)$, also s_A, t_A
- Morphisms $f=(f_0 : |A| \to |B|, f_1 : \Pi\, x,y.A(x,y) \to B(f_0(x), f_0(y)))$
- Set of morphisms forms a setoid $p:f \sim g$ is $p:\Pi\, x \in |A|.B(f_0(x),g_0(y))$

Worlds

- W category with pullbacks, spans can be completed to minimal pullbacks, all morphisms mono



Setoid-valued functors

- Setoid Aw for w∈ W
- Setoid morph Au:Aw→ Aw’ for u:w→ w’
  - Write u.a for Au(a)

A is *pullback-preserving* (p.p.f.) if for every pullback square $\mathsf{w}^{x}_{u}\Diamond^{x'}_{u'}\mathsf{w}'$ with apex $\overline{w}$ and low point $\underline{w}$ there is a continuous function of type

$$\Pi a \in A\mathsf{w}.\Pi a' \in A\mathsf{w}'.A\overline{w}(x.a, x'.a') \rightarrow \Sigma \underline{a} \in A\underline{w}.A\mathsf{w}(u.\underline{a}, a) \times A\mathsf{w}'(u'.\underline{a}, a')$$

So if values equal in apex, must have come from common value in intersection world

- Form category with morphisms given by pairs of functions, 2nd explicitly witnessing naturality up to ~
- For computation types actually use generalization called fibred setoids

Instantiations

- Axiomatised untyped model of stateful computation (soln of rec dom eqn) $\mathbb{L}$, $\mathbb{H}$,$\mathbb{V}$, $\mathbb{C}$
- Instantiation comprises
  - Category of worlds
  - Lluf subcatgory I of inclusions
  - Contravariant pushout preserving ς from W to setoids
  - Set of effects $\mathcal{E}$ and for each ϵ, set $\mathcal{R}(\epsilon)$ of relations on ς
- A relation on ς is admissible $Rw \subseteq \varsigma w \times \varsigma w$ st $(\sigma,\sigma') \in Rw$ and $u:w_0 \to w$ implies $(\sigma.u,\sigma'.u) \in Rw_0$ and closed under ~

Abstract Locations

An *abstract location* $\mathfrak{l}$ consists of:

- a nonempty, admissible PER $\mathfrak{l}^R$ on $\mathbb{H}$ modelling the "semantic equality" on the bits of the store that $\mathfrak{l}$ uses (a "rely-condition");
- a reflexive, transitive relation $\mathfrak{l}^G$ modelling what bits of the store "writes to $\mathfrak{l}$" leave intact (a "guarantee condition");
- an accessibility map $\mathfrak{l}^F : \Pi \mathsf{h} \in \mathbb{H}.\mathcal{P}(\mathrm{dom}(\mathsf{h}))$ describing the "footprint" of the abstract location.

subject to some accessibility conditions...

Two abstract locations $\mathfrak{l}_1, \mathfrak{l}_2$ are independent if

- for $i = 1, 2$ one has $\mathfrak{l}_i^R ; \mathfrak{l}_{3-i}^G \subseteq \mathfrak{l}_i^R$ and if $\mathfrak{l}_i^G(\mathsf{h}, \mathsf{h}_1)$ and $\mathsf{l} \in \mathrm{dom}(\mathsf{h}) \setminus \mathfrak{l}_{3-i}^F(\mathsf{h})$ then $\mathsf{l} \notin \mathfrak{l}_{3-i}^F(\mathsf{h}_1)$;
- If $(\mathsf{h}_1, \mathsf{h}_1) \in \mathfrak{l}_1$ and $(\mathsf{h}_2, \mathsf{h}_2) \in \mathfrak{l}_2$ there exists h such that $(\mathsf{h}, \mathsf{h}_1) \in \mathfrak{l}_1$ and $(\mathsf{h}, \mathsf{h}_2) \in \mathfrak{l}_2$.

If $\mathfrak{l}_1, \mathfrak{l}_2$ are independent, we form a joint location $\mathfrak{l}_1 \otimes \mathfrak{l}_2$ by $(\mathfrak{l}_1 \otimes \mathfrak{l}_2)^R = \mathfrak{l}_1^R \cap \mathfrak{l}_2^R$ and $(\mathfrak{l}_1 \otimes \mathfrak{l}_2)^G = (\mathfrak{l}_1^G \cup \mathfrak{l}_2^G)^*$ and $(\mathfrak{l}_1 \otimes \mathfrak{l}_2)^F(\mathsf{h}) = \mathfrak{l}_1^F(\mathsf{h}) \cup \mathfrak{l}_2^F(\mathsf{h})$.

Instantiation: Heap PERs

- World w is set of mutually independent abstract locations, tagged with regions

$$\mathfrak{S}\mathsf{w} = \{\mathsf{h} \in \mathbb{H} \mid \forall \mathfrak{l} \in \mathsf{w}.(\mathsf{h}, \mathsf{h}) \in \mathfrak{l}^R\}$$

$$\mathfrak{S}\mathsf{w}(\sigma, \sigma') = \{\star\} \iff \forall \mathfrak{l} \in \mathsf{w}.(\sigma, \sigma') \in \mathfrak{l}^R, \quad \mathfrak{S}\mathsf{w}(\sigma, \sigma') = \emptyset \text{ otherwise}$$

Morphism is injective u_0:dom w$\rightarrow$ dom w' plus pair of cts fns $\mathbb{H}\rightarrow\mathbb{H}$ that map the heaps between the two worlds in accordance with u_0 and play nice with relies & guarantees

Relations

$$
\begin{aligned}
R \in \mathcal{R}(rd_{\mathsf{r}}) \quad &\Longleftrightarrow \quad (\sigma, \sigma') \in R\mathsf{w} \Rightarrow \forall \mathfrak{I} \in \mathsf{w}(\mathsf{r}).(\sigma, \sigma') \in \mathfrak{I}^R \\
R \in \mathcal{R}(wr_{\mathsf{r}}) \quad &\Longleftrightarrow \quad (\sigma, \sigma') \in R\mathsf{w} \Rightarrow \forall \mathfrak{I} \in \mathsf{w}(\mathsf{r}).\forall (\sigma_1, \sigma_1') \in \mathfrak{I}^R \\
&\qquad (\sigma, \sigma_1) \in \mathfrak{I}^G \wedge (\sigma', \sigma_1') \in \mathfrak{I}^G \wedge \Rightarrow \\
&\qquad\qquad (\sigma_1, \sigma_1') \in R\mathsf{w} \\
R \in \mathcal{R}(al_{\mathsf{r}}) \quad &\Longleftrightarrow \quad (\sigma, \sigma') \in R\mathsf{w} \Rightarrow \forall \mathsf{w}_1.\forall u \in \mathcal{I}(\mathsf{w}, \mathsf{w}_1). \\
&\qquad (\mathsf{w}_1 \setminus \mathsf{w} \subseteq \mathsf{w}_1(\mathsf{r})) \Rightarrow \forall \sigma_1, \sigma_1' \in \mathfrak{S}\mathsf{w}_1. \\
&\qquad (\sigma_1.u \sim \sigma \wedge \sigma_1'.u \sim \sigma' \wedge \\
&\qquad (\sigma_1, \sigma_1') \in \textstyle\bigcap_{\mathfrak{I} \in \mathsf{w}_1 \setminus \mathsf{w}} \mathfrak{I}^R) \Rightarrow (\sigma_1, \sigma_1') \in R\mathsf{w}_1
\end{aligned}
$$

Proof-Relevant Logical Relations

A *semantic type* is a pair $(A, \Vdash^A)$ where A is a p.p.f. (on $\mathbf{W}$) and $\Vdash^A_{\mathsf{w}}$ is an admissible subset of $\mathbb{V} \times A\mathsf{w}$ for each $\mathsf{w} \in \mathbf{W}$ such that for every inclusion $u : \mathsf{w} \to \mathsf{w}'$ one has that $v \Vdash^A_{\mathsf{w}} \mathsf{v}$ implies $v \Vdash^A_{\mathsf{w}'} u.\mathsf{v}$. A *semantic computation* is a pair $(T, \Vdash^T)$ where T is a fibred setoid over $\mathbf{W}$ and $\Vdash^T_{\mathsf{w}}$ is an admissible subset of $\mathbb{C} \times T\mathsf{w}$ for each w.

Let A be a semantic type and ε an effect. A semantic computation $T_\varepsilon A$ is defined as follows:

- (Objects) Elements of $(T_\varepsilon A)\mathsf{w}$ are pairs $(\mathsf{c}_0, \mathsf{c}_1)$ of partial continuous functions where

$$\mathsf{c}_0 : \quad \mathfrak{S}\mathsf{w} \rightharpoonup \Sigma \mathsf{w}_1 . \mathcal{I}(\mathsf{w}, \mathsf{w}_1) \times \mathfrak{S}\mathsf{w}_1 \times A\mathsf{w}_1$$

and c_1 satisfies if $R \in \mathcal{R}(\varepsilon)$ and $(\sigma, \sigma') \in R\mathsf{w}$ and $\mathsf{c}_0(\sigma) = (\mathsf{w}_1, u, \sigma_1, a)$ and $\mathsf{c}_0(\sigma') = (\mathsf{w}'_1, u', \sigma'_1, a')$ then $\mathsf{c}_1(R, \sigma, \sigma')$ returns a pair $({}^{x}_{v}\Diamond^{x'}_{v'}, p)$ where $\mathsf{w}_1 \, {}^{x}_{v}\Diamond^{x'}_{v'} \mathsf{w}'_1$ such that $xu = x'u'$. Furthermore, $p \in A\overline{w}(x.a, x'.a')$ and, finally, $(\sigma_1.u, \sigma'_1.u') \in R\underline{\mathsf{w}}$ where $\underline{\mathsf{w}}$ and $\overline{\mathsf{w}}$ are low point and apex of ${}^{x}_{v}\Diamond^{x'}_{v'}$.

- (Proofs) ...

- (Realization) If $c \in \mathbb{C}$, define $c \Vdash^{T_\varepsilon A}_{\mathsf{w}} (\mathsf{c}_0, \mathsf{c}_1)$ to mean whenever $\mathsf{h} \Vdash_{\mathsf{w}} \sigma$ then $c(\mathsf{h})$ is defined iff $\mathsf{c}_0(\sigma)$ is defined and if $c(\mathsf{h}) = (\mathsf{h}_1, v)$ and $\mathsf{c}_0(\sigma) = (\mathsf{w}_1, u, \sigma_1, \mathsf{v})$ then $\mathsf{h}_1 \Vdash_{\mathsf{w}_1} \sigma_1$ and $v \Vdash^A_{\mathsf{w}_1} \mathsf{v}$.

Fundamental Theorem

To a typing derivation $\Gamma \vdash t : \tau \& \varepsilon$ we associate a morphism

$$[\![\Gamma \vdash t : \tau \& \varepsilon]\!] : S([\![\Gamma]\!]) \to T_\varepsilon [\![\tau]\!]$$

such that $[\![t]\!] \Vdash^{[\![\Gamma]\!] \to T_\varepsilon \tau} [\![\Gamma \vdash t : \tau \& \varepsilon]\!]$.

To an equality derivation $\Gamma \vdash t = t' : \tau \& \varepsilon$ we associate a proof

$$[\![\Gamma \vdash t = t' : \tau \& \varepsilon]\!] : [\![\Gamma \vdash t : \tau \& \varepsilon]\!] \sim [\![\Gamma \vdash t' : \tau \& \varepsilon]\!]$$

Theorem. If $(A, \Vdash^A)$ is a semantic type and $v \Vdash^A_{\mathsf{W}_0} e$ and $v' \Vdash^A_{\mathsf{W}_0} e'$ with $e \sim e'$ in A_{W_0} then v and v' are observationally equivalent at type A.

Payoff

- Usual equational theory holds in generic setting
- Some nu-calc examples in effect-refined versions
- All the effect-dependent equivalences we had before work in heap PERs instantiation, including masking
- Heap PERs can cope with multiplexed locations, giving read only effect to privately mutating set lookup, giving “pure function to pure function” type to memo functional, etc.
- Also some hard equivalences from the literature on models of local store