aulogo_u

Mechanized Logical Relations for Termination-Insensitive Noninterference

Simon O. Gregersen
joint work with Johan Bay, Amin Timany, and Lars Birkedal

Chalmers ProgLog/Security Seminar, 4th November 2020

Information-flow control tracks how information gets propagated through a program making sure the information is handled securely.

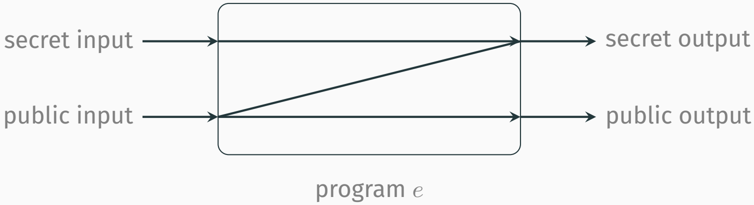
Explicit flow

$$l \leftarrow \,!h$$

Implicit flow

$$\begin{aligned}&l \leftarrow \textsf{false};\\&\textsf{if}\ \,!h\ \textsf{then}\ l \leftarrow \textsf{true}\end{aligned}$$

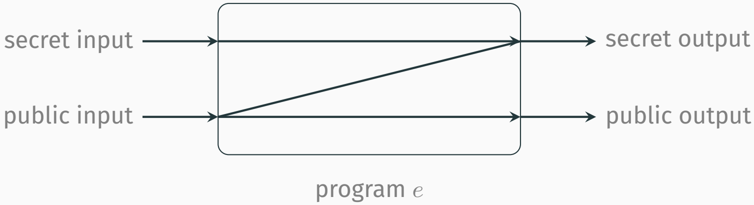
The prevailing basic semantic notion of secure information flow is **noninterference**.



Intuitively, **public outputs should be independent of secret inputs**: if e depends on a secret x then $NI(e)$ holds when

$$e[v_1/x] \Downarrow o_1 \quad \text{and} \quad e[v_2/x] \Downarrow o_2 \quad \text{implies} \quad o_1 \simeq o_2$$

The prevailing basic semantic notion of secure information flow is **noninterference**.



Intuitively, **public outputs should be independent of secret inputs**: if e depends on a secret x then $NI(e)$ holds when

$$e[v_1/x] \Downarrow o_1 \qquad \text{and} \qquad e[v_2/x] \Downarrow o_2 \qquad \text{implies} \qquad o_1 \simeq o_2$$

The problem

IFC enforcement is often specified using a static type system:

$$\Gamma \vdash e : t^{\ell} \qquad \text{implies} \qquad NI(e)$$

To be useful, it must support the same features as modern programming languages:

- higher types,
- reference types,
- abstract types,
- …

The difficulty of proving the type system sound, however, increases.

The problem

IFC enforcement is often specified using a static type system:

$$\Gamma \vdash e : t^{\ell} \qquad \text{implies} \qquad NI(e)$$

To be useful, it must support the same features as modern programming languages:

- higher types,
- reference types,
- abstract types,
- ...

The difficulty of proving the type system sound, however, increases.

This work

The main goal of this work is to

- show that such a rich type system satisfies **termination-insensitive noninterference**
- using a semantic model
  - $\Rightarrow$ compositional integration of syntactically well-typed and ill-typed components:

$$\Gamma, x : \tau_2 \vdash e_1 : \tau_1 \qquad \text{and} \qquad e_2 \in [\![\tau_2]\!] \qquad \text{then} \qquad TINI(e_1[e_2/x])$$

- with full mechanization of all results

This work

The main goal of this work is to

- show that such a rich type system satisfies **termination-insensitive noninterference**
- using a semantic model
  - $\Rightarrow$ compositional integration of syntactically well-typed and ill-typed components:

$$\Gamma, x : \tau_2 \vdash e_1 : \tau_1 \qquad \text{and} \qquad e_2 \in [\![\tau_2]\!] \qquad \text{then} \qquad \mathit{TINI}(e_1[e_2/x])$$

- with full mechanization of all results

Example (Multiplying by zero)

$$\lambda v.\, v * 0$$

cannot be syntactically typed at $\mathbb{N}^\top \to \mathbb{N}^\bot$.

Example (Temporary explicit leak)

$$\textsf{let}\ x = {!\,l}\ \textsf{in}\ l \leftarrow {!\,h};\ \ldots;\ l \leftarrow x$$

is not syntactically well-typed.

Example (Temporary implicit leak)

$$(\textsf{if}\ {!\,h}\ \textsf{then}\ l \leftarrow 1\ \textsf{else}\ l \leftarrow 0); l \leftarrow 0$$

is not syntactically well-typed.

Example (Multiplying by zero)

$$\lambda\, v.\, v * 0$$

cannot be syntactically typed at $\mathbb{N}^{\top} \to \mathbb{N}^{\perp}$.

Example (Temporary explicit leak)

$$\mathsf{let}\ x = !\, l\ \mathsf{in}\ l \leftarrow !\, h;\ \ldots;\ l \leftarrow x$$

is not syntactically well-typed.

Example (Temporary implicit leak)

$$(\mathsf{if}\ !\, h\ \mathsf{then}\ l \leftarrow 1\ \mathsf{else}\ l \leftarrow 0); l \leftarrow 0$$

is not syntactically well-typed.

Example (Multiplying by zero)

$$\lambda v.\, v * 0$$

cannot be syntactically typed at $\mathbb{N}^{\top} \to \mathbb{N}^{\perp}$.

Example (Temporary explicit leak)

$$\mathsf{let}\ x = !\, l\ \mathsf{in}\ l \leftarrow !\, h;\ \ldots;\ l \leftarrow x$$

is not syntactically well-typed.

Example (Temporary implicit leak)

$$(\mathsf{if}\ !\, h\ \mathsf{then}\ l \leftarrow 1\ \mathsf{else}\ l \leftarrow 0); l \leftarrow 0$$

is not syntactically well-typed.

Contributions

In summary, we address three major challenges:

- combining unary and binary models in the presence of higher-order state and impredicative polymorphism[1],
- constructing "logical" logical-relations models for termination-insensitive reasoning, while
- soundly allowing syntactically ill-typed but semantically secure programs to be composed with syntactically well-typed programs.

[1] Rajani and Garg (2018, 2020) sidestep all these difficulties by using syntactic worlds.

Contributions

In summary, we address three major challenges:

- combining unary and binary models in the presence of higher-order state and impredicative polymorphism[1],
- constructing "logical" logical-relations models for termination-insensitive reasoning, while
- soundly allowing syntactically ill-typed but semantically secure programs to be composed with syntactically well-typed programs.

[1] Rajani and Garg (2018, 2020) sidestep all these difficulties by using syntactic worlds.

Contributions

In summary, we address three major challenges:

- combining unary and binary models in the presence of higher-order state and impredicative polymorphism[1],
- constructing "logical" logical-relations models for termination-insensitive reasoning, while
- soundly allowing syntactically ill-typed but semantically secure programs to be composed with syntactically well-typed programs.

[1] Rajani and Garg (2018, 2020) sidestep all these difficulties by using syntactic worlds.

Language

$$
\begin{aligned}
e &::= \ \ldots \mid \lambda x.\, e \mid e\ e \mid \mathsf{ref}(e) \mid\ !\, e \mid e \leftarrow e \mid \Lambda\, e \mid \mathbb{\Lambda}\, e \mid e\ \_ \mid \\
&\qquad \mathsf{fold}\ \ e \mid \mathsf{unfold}\ \ e \mid \mathsf{pack}\, e \mid \mathsf{unpack}\, e\, \mathsf{as}\, x\, \mathsf{in}\, e \\
\ell &::= \kappa \mid l \in \mathcal{L} \mid \ell \sqcup \ell \\
\tau &::= t^{\ell} \\
t &::= \ \ldots \mid \tau \xrightarrow{\ell} \tau \mid \mathsf{ref}(\tau) \mid \alpha \mid \forall_{\ell}\, \alpha.\, \tau \mid \mathbb{\forall}_{\ell}\, \kappa.\, \tau \mid \exists \alpha.\, \tau \mid \mu\, \alpha.\, \tau
\end{aligned}
$$

Language

$$
\begin{aligned}
e &::= \ \ldots \mid \lambda x.\, e \mid e\ e \mid \mathsf{ref}(e) \mid\ !\, e \mid e \leftarrow e \mid \Lambda\, e \mid \mathbb{\Lambda}\, e \mid e\ \_ \mid \\
&\qquad \mathsf{fold}\ \ e \mid \mathsf{unfold}\ \ e \mid \mathsf{pack}\, e \mid \mathsf{unpack}\, e\, \mathsf{as}\, x\, \mathsf{in}\, e \\
\ell &::= \kappa \mid l \in \mathcal{L} \mid \ell \sqcup \ell \\
\tau &::= t^{\ell} \\
t &::= \ \ldots \mid \tau \xrightarrow{\ell} \tau \mid \mathsf{ref}(\tau) \mid \alpha \mid \forall_{\ell}\, \alpha.\, \tau \mid \mathbb{\forall}_{\ell}\, \kappa.\, \tau \mid \exists \alpha.\, \tau \mid \mu\, \alpha.\, \tau
\end{aligned}
$$

Consider if h then $f\ ()$ —if f has low side-effects we would leak h.

Typing judgment

Typing judgment

Typing judgment

Type-level context

Typing judgment

Type-level context

Label context

Typing judgment

Type-level context

Program counter label

Label context

Type system

T-NAT

$$\frac{n \in \mathbb{N}}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} n : \mathbb{N}^{\perp}}$$

T-BINOP

$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1 : \mathbb{N}^{\ell_1} \qquad \Xi \mid \Psi \mid \Gamma \vdash_{pc} e_2 : \mathbb{N}^{\ell_2} \qquad \circledcirc : \mathbb{N} \times \mathbb{N} \Rightarrow t}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1 \circledcirc e_2 : t^{\ell_1 \sqcup \ell_2}}$$

T-LAM

$$\frac{\Xi \mid \Psi \mid \Gamma, x : \tau_1 \vdash_{\ell_e} e : \tau_2}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \lambda x.\, e : \left(\tau_1 \xrightarrow{\ell_e} \tau_2\right)^{\perp}}$$

T-TLAM

$$\frac{\Xi, \alpha \mid \Psi \mid \Gamma \vdash_{\ell_e} e : \tau}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \Lambda\, e : \left(\forall_{\ell_e}\, \alpha.\, \tau\right)^{\perp}}$$

T-IF

$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : \mathbb{B}^{\ell} \qquad \forall i \in \{1, 2\}\,.\, \Xi \mid \Psi \mid \Gamma \vdash_{pc \sqcup \ell} e_i : \tau \qquad \Psi \vdash \tau \searrow \ell}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \textsf{if}\ e\ \textsf{then}\ e_1\ \textsf{else}\ e_2 : \tau}$$

T-STORE

$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1 : \textsf{ref}(\tau)^{\ell} \qquad \Xi \mid \Psi \mid \Gamma \vdash_{pc} e_2 : \tau \qquad \Psi \vdash \tau \searrow pc \sqcup \ell}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1 \leftarrow e_2 : 1^{\perp}}$$

Type system

T-NAT
$$\frac{n \in \mathbb{N}}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} n : \mathbb{N}^{\bot}}$$

T-BINOP
$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1 : \mathbb{N}^{\ell_1} \qquad \Xi \mid \Psi \mid \Gamma \vdash_{pc} e_2 : \mathbb{N}^{\ell_2} \qquad \circledcirc : \mathbb{N} \times \mathbb{N} \Rightarrow t}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1 \circledcirc e_2 : t^{\ell_1 \sqcup \ell_2}}$$

T-LAM
$$\frac{\Xi \mid \Psi \mid \Gamma, x : \tau_1 \vdash_{\ell_e} e : \tau_2}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \lambda x.\, e : \left(\tau_1 \xrightarrow{\ell_e} \tau_2\right)^{\bot}}$$

T-TLAM
$$\frac{\Xi, \alpha \mid \Psi \mid \Gamma \vdash_{\ell_e} e : \tau}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \Lambda\, e : \left(\forall_{\ell_e}\, \alpha.\, \tau\right)^{\bot}}$$

T-IF
$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : \mathbb{B}^{\ell} \qquad \forall i \in \{1, 2\}\,.\, \Xi \mid \Psi \mid \Gamma \vdash_{pc \sqcup \ell} e_i : \tau \qquad \Psi \vdash \tau \searrow \ell}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \textsf{if } e \textsf{ then } e_1 \textsf{ else } e_2 : \tau}$$

T-STORE
$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1 : \textsf{ref}(\tau)^{\ell} \qquad \Xi \mid \Psi \mid \Gamma \vdash_{pc} e_2 : \tau \qquad \Psi \vdash \tau \searrow pc \sqcup \ell}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1 \leftarrow e_2 : 1^{\bot}}$$

Type system

T-NAT
$$\frac{n \in \mathbb{N}}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} n : \mathbb{N}^{\bot}}$$

T-BINOP
$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1 : \mathbb{N}^{\ell_1} \qquad \Xi \mid \Psi \mid \Gamma \vdash_{pc} e_2 : \mathbb{N}^{\ell_2} \qquad \circledcirc : \mathbb{N} \times \mathbb{N} \Rightarrow t}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1 \circledcirc e_2 : t^{\ell_1 \sqcup \ell_2}}$$

T-LAM
$$\frac{\Xi \mid \Psi \mid \Gamma, x : \tau_1 \vdash_{\ell_e} e : \tau_2}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \lambda x.\, e : \left(\tau_1 \xrightarrow{\ell_e} \tau_2\right)^{\bot}}$$

T-TLAM
$$\frac{\Xi, \alpha \mid \Psi \mid \Gamma \vdash_{\ell_e} e : \tau}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \Lambda\, e : \left(\forall_{\ell_e} \alpha.\, \tau\right)^{\bot}}$$

T-IF
$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : \mathbb{B}^{\ell} \qquad \forall i \in \{1, 2\} \,.\, \Xi \mid \Psi \mid \Gamma \vdash_{pc \sqcup \ell} e_i : \tau \qquad \Psi \vdash \tau \searrow \ell}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \textsf{if}\ e\ \textsf{then}\ e_1\ \textsf{else}\ e_2 : \tau}$$

T-STORE
$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1 : \mathsf{ref}(\tau)^{\ell} \qquad \Xi \mid \Psi \mid \Gamma \vdash_{pc} e_2 : \tau \qquad \Psi \vdash \tau \searrow pc \sqcup \ell}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1 \leftarrow e_2 : 1^{\bot}}$$

Type system

T-NAT

$$\frac{n \in \mathbb{N}}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} n : \mathbb{N}^{\bot}}$$

T-BINOP

$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1 : \mathbb{N}^{\ell_1} \qquad \Xi \mid \Psi \mid \Gamma \vdash_{pc} e_2 : \mathbb{N}^{\ell_2} \qquad \circledcirc : \mathbb{N} \times \mathbb{N} \Rightarrow t}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1 \circledcirc e_2 : t^{\ell_1 \sqcup \ell_2}}$$

T-LAM

$$\frac{\Xi \mid \Psi \mid \Gamma, x : \tau_1 \vdash_{\ell_e} e : \tau_2}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \lambda x.\, e : \left(\tau_1 \xrightarrow{\ell_e} \tau_2\right)^{\bot}}$$

T-TLAM

$$\frac{\Xi, \alpha \mid \Psi \mid \Gamma \vdash_{\ell_e} e : \tau}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \Lambda\, e : \left(\forall_{\ell_e} \alpha.\, \tau\right)^{\bot}}$$

T-IF

$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : \mathbb{B}^{\ell} \qquad \forall i \in \{1,2\}\,.\, \Xi \mid \Psi \mid \Gamma \vdash_{pc \sqcup \ell} e_i : \tau \qquad \Psi \vdash \tau \searrow \ell}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \textsf{if}\ e\ \textsf{then}\ e_1\ \textsf{else}\ e_2 : \tau}$$

T-STORE

$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1 : \textsf{ref}(\tau)^{\ell} \qquad \Xi \mid \Psi \mid \Gamma \vdash_{pc} e_2 : \tau \qquad \Psi \vdash \tau \searrow pc \sqcup \ell}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1 \leftarrow e_2 : 1^{\bot}}$$

Theorem (Termination-Insensitive Noninterference)

Given $\bot \sqsubseteq \top$ *and* $\top \not\sqsubseteq \bot$*, if*

$$\cdot \mid \cdot \mid x : \mathbb{B}^{\top} \vdash_{\bot} e : \mathbb{B}^{\bot},$$
$$\cdot \mid \cdot \mid \cdot \vdash_{\bot} v_1 : \mathbb{B}^{\top}, \textit{ and } \cdot \mid \cdot \mid \cdot \vdash_{\bot} v_2 : \mathbb{B}^{\top}$$

then

$$(\emptyset, e[v_1/x]) \to^{*} (\sigma_1, v_1') \textit{ and } (\emptyset, e[v_2/x]) \to^{*} (\sigma_2, v_2') \textit{ then } v_1' = v_2'.$$

Theorem (Termination-Insensitive Noninterference)

Given $\bot \sqsubseteq \top$ *and* $\top \not\sqsubseteq \bot$*, if*

$$\cdot \mid \cdot \mid x : \mathbb{B}^{\top} \vdash_{\bot} e : \mathbb{B}^{\bot},$$

$$\cdot \mid \cdot \mid \cdot \vdash_{\bot} v_1 : \mathbb{B}^{\top}, \text{ and } \cdot \mid \cdot \mid \cdot \vdash_{\bot} v_2 : \mathbb{B}^{\top}$$

then

$$(\emptyset, e[v_1/x]) \rightarrow^* (\sigma_1, v_1') \text{ and } (\emptyset, e[v_2/x]) \rightarrow^* (\sigma_2, v_2') \text{ then } v_1' = v_2'.$$

Theorem (Termination-Insensitive Noninterference)

Given $\bot \sqsubseteq \top$ *and* $\top \not\sqsubseteq \bot$*, if*

$$\cdot \mid \cdot \mid x : \mathbb{B}^{\top} \vdash_{\bot} e : \mathbb{B}^{\bot},$$

$$\cdot \mid \cdot \mid \cdot \vdash_{\bot} v_1 : \mathbb{B}^{\top}, \textit{ and } \cdot \mid \cdot \mid \cdot \vdash_{\bot} v_2 : \mathbb{B}^{\top}$$

then

$$(\emptyset, e[v_1/x]) \rightarrow^{*} (\sigma_1, v_1') \textit{ and } (\emptyset, e[v_2/x]) \rightarrow^{*} (\sigma_2, v_2') \textit{ then } v_1' = v_2'.$$

Theorem (Termination-Insensitive Noninterference)

Given $\bot \sqsubseteq \top$ *and* $\top \not\sqsubseteq \bot$, *if*

$$\cdot \mid \cdot \mid x : \mathbb{B}^{\top} \vdash_{\bot} e : \mathbb{B}^{\bot},$$
$$\cdot \mid \cdot \mid \cdot \vdash_{\bot} v_1 : \mathbb{B}^{\top}, \textit{ and } \cdot \mid \cdot \mid \cdot \vdash_{\bot} v_2 : \mathbb{B}^{\top}$$

then

$$(\emptyset, e[v_1/x]) \to^* (\sigma_1, v_1') \textit{ and } (\emptyset, e[v_2/x]) \to^* (\sigma_2, v_2') \textit{ then } v_1' = v_2'.$$

What we want to do

We set up a (logical) relation

$$\Xi \mid \Psi \mid \Gamma \vDash e_1 \approx e_2 : \tau$$

such that

$$\begin{aligned} \Xi \mid \Psi \mid \Gamma \vdash_{pc} e : \tau \qquad &\Rightarrow \qquad \Xi \mid \Psi \mid \Gamma \vDash e \approx e : \tau \\ \Xi \mid \Psi \mid \Gamma \vDash e \approx e : \tau \qquad &\Rightarrow \qquad \mathit{TINI}(e) \end{aligned}$$

However, this requires manipulating and defining step-indexed Kripke models over recursive worlds which induces a lot of complexity.

What we do

We combat this complexity by using Iris and *iProp* for defining our semantic domain.

- The later modality ($\triangleright$) to reason about step-indices,
- User-definable ghost resources,
- The update modality ($\Rrightarrow$) for reasoning about ghost resources, and
- We can use the Coq formalization and IPM to mechanize our proofs.

While hiding details, we still have to think in terms of step-indices and updates …

What we do

We combat this complexity by using Iris and *iProp* for defining our semantic domain.

- The later modality ($\triangleright$) to reason about step-indices,
- User-definable ghost resources,
- The update modality ($\Rrightarrow$) for reasoning about ghost resources, and
- We can use the Coq formalization and IPM to mechanize our proofs.

While hiding details, we still have to think in terms of step-indices and updates ...

What we do cont'd

Existing works on defining "logical" logical relations are aimed at proving (contextual) refinements: Intuitively, e_1 refines e_2 if

$$e_1 \Downarrow v_1 \Rightarrow e_2 \Downarrow v_2 \wedge v_1 \approx v_2.$$

This can expressed using Iris-style weakest precondition predicates, i.e.,

$$\mathsf{wp}\ e_1\ \{v_1.\ e_2 \Downarrow v_2 \wedge v_1 \approx v_2\}.$$

However, we need a termination-insensitive notion:

$$e_1 \Downarrow v_1 \wedge e_2 \Downarrow v_2 \Rightarrow v_1 \approx v_2.$$

What we do cont'd

Existing works on defining "logical" logical relations are aimed at proving (contextual) refinements: Intuitively, e_1 refines e_2 if

$$e_1 \Downarrow v_1 \Rightarrow e_2 \Downarrow v_2 \land v_1 \approx v_2.$$

This can expressed using Iris-style weakest precondition predicates, i.e.,

$$\mathsf{wp}\; e_1 \; \{v_1.\; e_2 \Downarrow v_2 \land v_1 \approx v_2\}.$$

However, we need a termination-insensitive notion:

$$e_1 \Downarrow v_1 \land e_2 \Downarrow v_2 \Rightarrow v_1 \approx v_2.$$

What we do cont'd

Existing works on defining "logical" logical relations are aimed at proving (contextual) refinements: Intuitively, e_1 refines e_2 if

$$e_1 \Downarrow v_1 \Rightarrow e_2 \Downarrow v_2 \wedge v_1 \approx v_2.$$

This can expressed using Iris-style weakest precondition predicates, i.e.,

$$\mathsf{wp}\; e_1 \; \{v_1.\; e_2 \Downarrow v_2 \wedge v_1 \approx v_2\}.$$

However, we need a termination-insensitive notion:

$$e_1 \Downarrow v_1 \wedge e_2 \Downarrow v_2 \Rightarrow v_1 \approx v_2.$$

Nested weakest preconditions do not work ...

Nested weakest preconditions, *e.g.*, using

$$e_1 \approx e_2 \triangleq \mathsf{wp}\ e_1\ \{v_1.\ \mathsf{wp}\ e_2\ \{v_2.\ v_1 \approx v_2\}\}$$

does not admit a strong enough bind rule:

$$\dfrac{\dfrac{\mathsf{wp}\ e\ \{w.\ \mathsf{wp}\ K[w]\ \{v.\ \mathsf{wp}\ e'\ \{w'.\ \mathsf{wp}\ K'[w']\ \{v'.\ v \approx v'\}\}\}\}}{\mathsf{wp}\ e\ \{w.\ \mathsf{wp}\ K[w]\ \{v.\ \mathsf{wp}\ K'[e']\ \{v'.\ v \approx v'\}\}\}}\ \textsc{wp-mono} + \textsc{wp-bind}}{\mathsf{wp}\ K[e]\ \{v.\ \mathsf{wp}\ K'[e']\ \{v'.\ v \approx v'\}\}}\ \textsc{wp-bind}$$

Semantic model

Semantic model

Our semantic model formalizes an **observer-sensitive equivalence**

$$\Xi \,|\, \Psi \,|\, \Gamma \vDash e \approx_{\zeta} e' : \tau$$

for any $\zeta \in \mathcal{L}$.

Theorem (Binary fundamental theorem)

If $\quad \Xi \,|\, \Psi \,|\, \Gamma \vdash_{pc} e : \tau \quad$ *then* $\quad \forall \zeta. \, \Xi \,|\, \Psi \,|\, \Gamma \vDash e \approx_{\zeta} e : \tau.$

Semantic model cont'd

A central idea in the model is to interpret types both as a

Binary relation for relating terms that are observationally equivalent and as a

Unary relation for characterizing terms that have no "illegal" side-effects.

Semantic model cont'd

A central idea in the model is to interpret types both as a

Binary relation for relating terms that are observationally equivalent and as a

Unary relation for characterizing terms that have no "illegal" side-effects.

If $\top \not\sqsubseteq \zeta$ and

$$e : t^{\top} \qquad \text{and} \qquad e' : t^{\top}$$

then the programs can—individually—do "whatever they feel like" while being observationally equivalent as long as they do not have observable side-effects.

Semantic model cont'd

A central idea in the model is to interpret types both as a

Binary relation for relating terms that are observationally equivalent and as a

Unary relation for characterizing terms that have no "illegal" side-effects.

If $\top \not\sqsubseteq \zeta$ and

$$e : t^{\top} \qquad \text{and} \qquad e' : t^{\top}$$

then the programs can—individually—do "whatever they feel like" while being observationally equivalent as long as they do not have observable side-effects.

The logical relation

We define both unary and binary variants of

- an expression relation $\mathcal{E}[\![\tau]\!]^{\rho}_{\Theta}$ for closed expressions, and
- a value relation $[\![\tau]\!]^{\rho}_{\Theta}$

as is custom for logical relations models. From these,

$$\Xi \mid \Psi \mid \Gamma \vDash e \approx_{\zeta} e' : \tau$$

follows by closing with well-typed substitutions.

Value relations

We define

$$\llbracket \tau \rrbracket^{\rho}_{\Theta} : \mathit{Val} \times \mathit{Val} \to \mathit{iProp} \qquad \llbracket \tau \rrbracket^{\rho}_{\Delta} : \mathit{Val} \to \mathit{iProp}$$

$$\llbracket t \rrbracket^{\rho}_{\Theta} : \mathit{Val} \times \mathit{Val} \to \mathit{iProp} \qquad \llbracket t \rrbracket^{\rho}_{\Delta} : \mathit{Val} \to \mathit{iProp}$$

where

$$\rho : \mathit{LabelVar} \to \mathcal{L}$$

$$\Theta : \mathit{TypeVar} \to \mathit{Rel} \times \mathit{Pred} \times \mathit{Pred}$$

$$\Delta : \mathit{TypeVar} \to \mathit{Pred}$$

given $\mathit{Rel} \triangleq \mathit{Val} \times \mathit{Val} \to \mathit{iProp}$ and $\mathit{Pred} \triangleq \mathit{Val} \to \mathit{iProp}$.

Binary-unary subsumption property

It will be crucial that

$$\forall v, v'.\, [\![\tau]\!]^{\rho}_{\Theta}(v, v') \twoheadrightarrow [\![\tau]\!]^{\rho}_{\Theta_L}(v) * [\![\tau]\!]^{\rho}_{\Theta_R}(v')$$

holds where $\Theta_L \triangleq \pi_2 \circ \Theta$ and $\Theta_R \triangleq \pi_3 \circ \Theta$.

When interpreting type variables,

$$[\![\alpha]\!]^{\rho}_{\Theta} \triangleq \pi_1\left(\Theta(\alpha)\right)$$

this requires that Θ is *coherent*:

$$Coh(\Theta) \triangleq \mathop{\Huge *}_{(\Phi, \Phi_L, \Phi_R) \in \mathrm{Im}(\Theta)} \Box \left(\forall v, v'.\, \Phi(v, v') \twoheadrightarrow \Phi_L(v) * \Phi_R(v')\right).$$

Binary-unary subsumption property

It will be crucial that

$$\forall v, v'.\, [\![\tau]\!]^{\rho}_{\Theta}(v, v') \twoheadrightarrow [\![\tau]\!]^{\rho}_{\Theta_L}(v) * [\![\tau]\!]^{\rho}_{\Theta_R}(v')$$

holds where $\Theta_L \triangleq \pi_2 \circ \Theta$ and $\Theta_R \triangleq \pi_3 \circ \Theta$.

When interpreting type variables,

$$[\![\alpha]\!]^{\rho}_{\Theta} \triangleq \pi_1\left(\Theta(\alpha)\right)$$

this requires that Θ is *coherent*:

$$Coh(\Theta) \triangleq \mathop{\Huge *}_{(\Phi, \Phi_L, \Phi_R) \in \mathrm{Im}(\Theta)} \square \left(\forall v, v'.\, \Phi(v, v') \twoheadrightarrow \Phi_L(v) * \Phi_R(v')\right).$$

Binary value interpretation of labeled types

To interpret labeled types, we make use of an interpretation of syntactic labels.

$$\begin{aligned}
[\![\cdot]\!]_\rho \;\; &: \; \mathit{Label}_{\mathcal{L}} \to \mathcal{L} \\
[\![\kappa]\!]_\rho &\triangleq \rho(\kappa) \\
[\![l]\!]_\rho &\triangleq l \\
[\![\ell_1 \sqcup \ell_2]\!]_\rho &\triangleq [\![\ell_1]\!]_\rho \sqcup [\![\ell_2]\!]_\rho.
\end{aligned}$$

This allows us to formally express the intuition given previously:

$$[\![t^\ell]\!]^\rho_\Theta(v, v') \triangleq \begin{cases} [\![t]\!]^\rho_\Theta(v, v') & \text{if } [\![\ell]\!]_\rho \sqsubseteq \zeta \\ [\![t]\!]^\rho_{\Theta_L}(v) * [\![t]\!]^\rho_{\Theta_R}(v') & \text{if } [\![\ell]\!]_\rho \not\sqsubseteq \zeta \end{cases}$$

Binary value interpretation of unlabeled types

Mostly standard, but we have to make sure the subsumption property holds.

$$\llbracket \mathbb{B} \rrbracket^{\rho}_{\Theta}(v, v') \triangleq v = v' \in \{\mathsf{true}, \mathsf{false}\}$$

$$\llbracket \tau_1 \xrightarrow{\ell_e} \tau_2 \rrbracket^{\rho}_{\Theta}(v, v') \triangleq \Box\left(\forall w, w'.\, \llbracket \tau_1 \rrbracket^{\rho}_{\Theta}(w, w') \mathrel{-\!\!*} \mathcal{E}\llbracket \tau_2 \rrbracket^{\rho}_{\Theta}(v\ w, v'\ w')\right) * \llbracket \tau_1 \xrightarrow{\ell_e} \tau_2 \rrbracket^{\rho}_{\Theta_L}(v) * \llbracket \tau_1 \xrightarrow{\ell_e} \tau_2 \rrbracket^{\rho}_{\Theta_R}(v')$$

$$\llbracket \mathsf{ref}(\tau) \rrbracket^{\rho}_{\Theta}(v, v') \triangleq \exists \iota, \iota'.\, v = \iota * v' = \iota' * \boxed{\exists w, w'.\, \iota \mapsto_L w * \iota' \mapsto_R w' * \llbracket \tau \rrbracket^{\rho}_{\Theta}(w, w')}^{\mathcal{N}_{root}.(\iota, \iota')}$$

$$\llbracket \alpha \rrbracket^{\rho}_{\Theta} \triangleq \pi_1\left(\Theta(\alpha)\right)$$

$$\llbracket \forall_{\ell_e}\, \alpha.\, \tau \rrbracket^{\rho}_{\Theta}(v, v') \triangleq \Box\Big(\forall \Phi : \mathit{Rel}.\, \forall \Phi_L, \Phi_R : \mathit{Pred}.\, \Box\left(\forall v, v'.\, \Phi(v, v') \mathrel{-\!\!*} \Phi_L(v) * \Phi_R(v')\right) \mathrel{-\!\!*} \mathcal{E}\llbracket \tau \rrbracket^{\rho}_{\Theta, \alpha \mapsto (\Phi, \Phi_L, \Phi_R)}(v\ \_, v'\ \_)\Big) * \llbracket \forall_{\ell_e}\, \alpha.\, \tau \rrbracket^{\rho}_{\Theta_L}(v) * \llbracket \forall_{\ell_e}\, \alpha.\, \tau \rrbracket^{\rho}_{\Theta_R}(v')$$

$$\llbracket \forall_{\ell_e}\, \kappa.\, \tau \rrbracket^{\rho}_{\Theta}(v, v') \triangleq \Box\left(\forall l \in \mathcal{L}.\, \mathcal{E}\llbracket \tau \rrbracket^{\rho, \kappa \mapsto l}_{\Theta}(v\ \_, v'\ \_)\right) * \llbracket \forall_{\ell_e}\, \kappa.\, \tau \rrbracket^{\rho}_{\Theta_L}(v) * \llbracket \forall_{\ell_e}\, \kappa.\, \tau \rrbracket^{\rho}_{\Theta_R}(v')$$

Binary value interpretation of unlabeled types

Mostly standard, but we have to make sure the subsumption property holds.

$$\llbracket \mathbb{B} \rrbracket_{\Theta}^{\rho}(v, v') \triangleq v = v' \in \{\mathsf{true}, \mathsf{false}\}$$

$$\llbracket \tau_1 \xrightarrow{\ell_e} \tau_2 \rrbracket_{\Theta}^{\rho}(v, v') \triangleq \Box\,(\forall w, w'.\, \llbracket \tau_1 \rrbracket_{\Theta}^{\rho}(w, w') \mathrel{-\!\!*} \mathcal{E}\llbracket \tau_2 \rrbracket_{\Theta}^{\rho}(v\ w, v'\ w')) * \llbracket \tau_1 \xrightarrow{\ell_e} \tau_2 \rrbracket_{\Theta_L}^{\rho}(v) * \llbracket \tau_1 \xrightarrow{\ell_e} \tau_2 \rrbracket_{\Theta_R}^{\rho}(v')$$

$$\llbracket \mathsf{ref}(\tau) \rrbracket_{\Theta}^{\rho}(v, v') \triangleq \exists \iota, \iota'.\, v = \iota * v' = \iota' * \boxed{\exists w, w'.\, \iota \mapsto_L w * \iota' \mapsto_R w' * \llbracket \tau \rrbracket_{\Theta}^{\rho}(w, w')}^{\mathcal{N}_{root}.(\iota, \iota')}$$

$$\llbracket \alpha \rrbracket_{\Theta}^{\rho} \triangleq \pi_1\,(\Theta(\alpha))$$

$$\llbracket \forall_{\ell_e}\, \alpha.\, \tau \rrbracket_{\Theta}^{\rho}(v, v') \triangleq \Box\,\Big(\forall \Phi : \textit{Rel}.\, \forall \Phi_L, \Phi_R : \textit{Pred}.\, \Box\,(\forall v, v'.\, \Phi(v, v') \mathrel{-\!\!*} \Phi_L(v) * \Phi_R(v')) \mathrel{-\!\!*} \mathcal{E}\llbracket \tau \rrbracket_{\Theta, \alpha \mapsto (\Phi, \Phi_L, \Phi_R)}^{\rho}(v\ \_, v'\ \_)\Big) * \llbracket \forall_{\ell_e}\, \alpha.\, \tau \rrbracket_{\Theta_L}^{\rho}(v) * \llbracket \forall_{\ell_e}\, \alpha.\, \tau \rrbracket_{\Theta_R}^{\rho}(v')$$

$$\llbracket \forall_{\ell_e}\, \kappa.\, \tau \rrbracket_{\Theta}^{\rho}(v, v') \triangleq \Box\,\Big(\forall l \in \mathcal{L}.\, \mathcal{E}\llbracket \tau \rrbracket_{\Theta}^{\rho, \kappa \mapsto l}(v\ \_, v'\ \_)\Big) * \llbracket \forall_{\ell_e}\, \kappa.\, \tau \rrbracket_{\Theta_L}^{\rho}(v) * \llbracket \forall_{\ell_e}\, \kappa.\, \tau \rrbracket_{\Theta_R}^{\rho}(v')$$

Binary value interpretation of unlabeled types

Mostly standard, but we have to make sure the subsumption property holds.

$$[\![\mathbb{B}]\!]^{\rho}_{\Theta}(v, v') \triangleq v = v' \in \{\mathsf{true}, \mathsf{false}\}$$

$$[\![\tau_1 \xrightarrow{\ell_e} \tau_2]\!]^{\rho}_{\Theta}(v, v') \triangleq \square\,(\forall w, w'.\, [\![\tau_1]\!]^{\rho}_{\Theta}(w, w') \mathrel{-\!\!*} \mathcal{E}[\![\tau_2]\!]^{\rho}_{\Theta}(v\ w, v'\ w')) \ast$$
$$[\![\tau_1 \xrightarrow{\ell_e} \tau_2]\!]^{\rho}_{\Theta_L}(v) \ast [\![\tau_1 \xrightarrow{\ell_e} \tau_2]\!]^{\rho}_{\Theta_R}(v')$$

$$[\![\mathsf{ref}(\tau)]\!]^{\rho}_{\Theta}(v, v') \triangleq \exists \iota, \iota'.\, v = \iota \ast v' = \iota' \ast \boxed{\exists w, w'.\, \iota \mapsto_L w \ast \iota' \mapsto_R w' \ast [\![\tau]\!]^{\rho}_{\Theta}(w, w')}^{\mathcal{N}_{root}.(\iota, \iota')}$$

$$[\![\alpha]\!]^{\rho}_{\Theta} \triangleq \pi_1\,(\Theta(\alpha))$$

$$[\![\forall_{\ell_e}\, \alpha.\, \tau]\!]^{\rho}_{\Theta}(v, v') \triangleq \square\,\Big(\forall \Phi : \mathit{Rel}.\, \forall \Phi_L, \Phi_R : \mathit{Pred}.$$
$$\square\,(\forall v, v'.\, \Phi(v, v') \mathrel{-\!\!*} \Phi_L(v) \ast \Phi_R(v')) \mathrel{-\!\!*} \mathcal{E}[\![\tau]\!]^{\rho}_{\Theta, \alpha \mapsto (\Phi, \Phi_L, \Phi_R)}(v\ \_, v'\ \_)\Big) \ast$$
$$[\![\forall_{\ell_e}\, \alpha.\, \tau]\!]^{\rho}_{\Theta_L}(v) \ast [\![\forall_{\ell_e}\, \alpha.\, \tau]\!]^{\rho}_{\Theta_R}(v')$$

$$[\![\forall_{\ell_e}\, \kappa.\, \tau]\!]^{\rho}_{\Theta}(v, v') \triangleq \square\,\Big(\forall l \in \mathcal{L}.\, \mathcal{E}[\![\tau]\!]^{\rho, \kappa \mapsto l}_{\Theta}(v\ \_, v'\ \_)\Big) \ast [\![\forall_{\ell_e}\, \kappa.\, \tau]\!]^{\rho}_{\Theta_L}(v) \ast [\![\forall_{\ell_e}\, \kappa.\, \tau]\!]^{\rho}_{\Theta_R}(v')$$

Binary value interpretation of unlabeled types

Mostly standard, but we have to make sure the subsumption property holds.

$$[\![\mathbb{B}]\!]^{\rho}_{\Theta}(v, v') \triangleq v = v' \in \{\textsf{true}, \textsf{false}\}$$

$$[\![\tau_1 \xrightarrow{\ell_e} \tau_2]\!]^{\rho}_{\Theta}(v, v') \triangleq \Box\,(\forall w, w'.\, [\![\tau_1]\!]^{\rho}_{\Theta}(w, w') \twoheadrightarrow \mathcal{E}[\![\tau_2]\!]^{\rho}_{\Theta}(v\ w, v'\ w')) \ast$$
$$[\![\tau_1 \xrightarrow{\ell_e} \tau_2]\!]^{\rho}_{\Theta_L}(v) \ast [\![\tau_1 \xrightarrow{\ell_e} \tau_2]\!]^{\rho}_{\Theta_R}(v')$$

$$[\![\textsf{ref}(\tau)]\!]^{\rho}_{\Theta}(v, v') \triangleq \exists \iota, \iota'.\, v = \iota \ast v' = \iota' \ast \boxed{\exists w, w'.\, \iota \mapsto_L w \ast \iota' \mapsto_R w' \ast [\![\tau]\!]^{\rho}_{\Theta}(w, w')}^{\mathcal{N}_{root}.(\iota,\iota')}$$

$$[\![\alpha]\!]^{\rho}_{\Theta} \triangleq \pi_1\,(\Theta(\alpha))$$

$$[\![\forall_{\ell_e}\, \alpha.\, \tau]\!]^{\rho}_{\Theta}(v, v') \triangleq \Box\,\Big(\forall \Phi : \textit{Rel}.\, \forall \Phi_L, \Phi_R : \textit{Pred}.$$
$$\Box\,(\forall v, v'.\, \Phi(v, v') \twoheadrightarrow \Phi_L(v) \ast \Phi_R(v')) \twoheadrightarrow \mathcal{E}[\![\tau]\!]^{\rho}_{\Theta, \alpha \mapsto (\Phi, \Phi_L, \Phi_R)}(v\ \_, v'\ \_)\Big) \ast$$
$$[\![\forall_{\ell_e}\, \alpha.\, \tau]\!]^{\rho}_{\Theta_L}(v) \ast [\![\forall_{\ell_e}\, \alpha.\, \tau]\!]^{\rho}_{\Theta_R}(v')$$

$$[\![\forall_{\ell_e}\, \kappa.\, \tau]\!]^{\rho}_{\Theta}(v, v') \triangleq \Box\,\Big(\forall l \in \mathcal{L}.\, \mathcal{E}[\![\tau]\!]^{\rho, \kappa \mapsto l}_{\Theta}(v\ \_, v'\ \_)\Big) \ast [\![\forall_{\ell_e}\, \kappa.\, \tau]\!]^{\rho}_{\Theta_L}(v) \ast [\![\forall_{\ell_e}\, \kappa.\, \tau]\!]^{\rho}_{\Theta_R}(v')$$

Binary value interpretation of unlabeled types

Mostly standard, but we have to make sure the subsumption property holds.

$$[\![\mathbb{B}]\!]^{\rho}_{\Theta}(v, v') \triangleq v = v' \in \{\mathsf{true}, \mathsf{false}\}$$

$$[\![\tau_1 \xrightarrow{\ell_e} \tau_2]\!]^{\rho}_{\Theta}(v, v') \triangleq \square\, (\forall w, w'.\, [\![\tau_1]\!]^{\rho}_{\Theta}(w, w') \twoheadrightarrow \mathcal{E}[\![\tau_2]\!]^{\rho}_{\Theta}(v\ w, v'\ w')) * [\![\tau_1 \xrightarrow{\ell_e} \tau_2]\!]^{\rho}_{\Theta_L}(v) * [\![\tau_1 \xrightarrow{\ell_e} \tau_2]\!]^{\rho}_{\Theta_R}(v')$$

$$[\![\mathsf{ref}(\tau)]\!]^{\rho}_{\Theta}(v, v') \triangleq \exists \iota, \iota'.\, v = \iota * v' = \iota' * \boxed{\exists w, w'.\, \iota \mapsto_L w * \iota' \mapsto_R w' * [\![\tau]\!]^{\rho}_{\Theta}(w, w')}^{\mathcal{N}_{root}.(\iota, \iota')}$$

$$[\![\alpha]\!]^{\rho}_{\Theta} \triangleq \pi_1\, (\Theta(\alpha))$$

$$[\![\forall_{\ell_e}\, \alpha.\, \tau]\!]^{\rho}_{\Theta}(v, v') \triangleq \square\, \Big(\forall \Phi : \mathit{Rel}.\, \forall \Phi_L, \Phi_R : \mathit{Pred}.\, \square\, (\forall v, v'.\, \Phi(v, v') \twoheadrightarrow \Phi_L(v) * \Phi_R(v')) \twoheadrightarrow \mathcal{E}[\![\tau]\!]^{\rho}_{\Theta, \alpha \mapsto (\Phi, \Phi_L, \Phi_R)}(v\ \_, v'\ \_)\Big) * [\![\forall_{\ell_e}\, \alpha.\, \tau]\!]^{\rho}_{\Theta_L}(v) * [\![\forall_{\ell_e}\, \alpha.\, \tau]\!]^{\rho}_{\Theta_R}(v')$$

$$[\![\forall_{\ell_e}\, \kappa.\, \tau]\!]^{\rho}_{\Theta}(v, v') \triangleq \square \left(\forall l \in \mathcal{L}.\, \mathcal{E}[\![\tau]\!]^{\rho, \kappa \mapsto l}_{\Theta}(v\ \_, v'\ \_)\right) * [\![\forall_{\ell_e}\, \kappa.\, \tau]\!]^{\rho}_{\Theta_L}(v) * [\![\forall_{\ell_e}\, \kappa.\, \tau]\!]^{\rho}_{\Theta_R}(v')$$

Modal weakest preconditions

We develop a theory of **modal weakest preconditions**

$$\mathsf{mwp}^{\mathcal{M};a}\, e\, \{\Phi\}$$

with the intuitive meaning

$$\forall \sigma, \sigma', v.\, (e, \sigma) \to^* (v, \sigma') \twoheadrightarrow \mathcal{M}(\Phi(v)).$$

With a valid modality $\mathcal{M}$, the connective admits several general structural rules

MWP instances

Crucially, we get a unary connective $\mathsf{mwp}^{\mathcal{M}\Rrightarrow\triangleright}\, e\,\{\Phi\}$ that implies

$$\forall \sigma, \sigma', v.\, (e, \sigma) \rightarrow^* (v, \sigma') \twoheadrightarrow \Phi(v)$$

and a binary connective $\mathsf{mwp}\; e_1 \sim e_2\,\{\Phi\}$ that implies

$$\begin{aligned} &\forall \sigma_1, \sigma_1', v.\, (e_1, \sigma_1) \rightarrow^* (v, \sigma_1') \twoheadrightarrow \\ &\qquad \forall \sigma_2, \sigma_2', w.\, (e_2, \sigma_2) \rightarrow^* (w, \sigma_2') \twoheadrightarrow \Phi(v, w) \end{aligned}$$

MWP instances cont'd

Lemma (Binary MWP - bind)

$$\frac{\text{mwp}\; e \sim e' \; \{v, v'.\ \text{mwp}\; K[v] \sim K'[v'] \; \{\varPhi\}\}}{\text{mwp}\; K[e] \sim K'[e'] \; \{\varPhi\}}$$

Expression relations

We can now define the binary expression relation

$$\mathcal{E}[\![\tau]\!]^{\rho}_{\Theta}(e, e') \triangleq \mathsf{mwp}\; e_1 \sim e_2 \,\{[\![\tau]\!]^{\rho}_{\Theta}\}$$

as well as the unary expression relation

$$\mathcal{E}_{pc}[\![\tau]\!]^{\rho}_{\Delta}(e) \triangleq [\![pc]\!]_{\rho} \not\sqsubseteq \zeta \Rightarrow \mathsf{mwp}^{\mathcal{M} \Rrightarrow\!\triangleright}\; e \,\{[\![\tau]\!]^{\rho}_{\Delta}\}.$$

More Examples

Static semantic typing instead of dynamic enforcement

Fennel and Thiemann (2013) consider a report processing application with

$$\textit{sendToManager} : \mathsf{ref}(\textit{Report}^{\top}) \xrightarrow{\top} 1$$

$$\textit{sendToFacebook} : \mathsf{ref}(\textit{Report}^{\bot}) \xrightarrow{\bot} 1$$

with the extension

$$\begin{aligned} \textit{addPrivileged} \triangleq \lambda\, & isPrivileged, worker, report. \\ & \mathsf{if}\ isPrivileged\ \mathsf{then}\ report \leftarrow !\, report + !\, h\ \mathsf{else}\ () \\ & worker\ \ report \end{aligned}$$

addPrivileged true *sendToManager* syntactically type checks but
addPrivileged false *sendToFacebook* does not.

Static semantic typing instead of dynamic enforcement

Fennel and Thiemann (2013) consider a report processing application with

$$\mathit{sendToManager} : \mathsf{ref}(\mathit{Report}^{\top}) \xrightarrow{\top} 1$$
$$\mathit{sendToFacebook} : \mathsf{ref}(\mathit{Report}^{\bot}) \xrightarrow{\bot} 1$$

with the extension

$$\begin{aligned}\mathit{addPrivileged} \triangleq \lambda\, & isPrivileged, worker, report. \\ & \textsf{if } isPrivileged \textsf{ then } report \leftarrow !\,report + !\,h \textsf{ else } () \\ & worker\ report\end{aligned}$$

addPrivileged true *sendToManager* syntactically type checks but *addPrivileged* false *sendToFacebook* does not.

Static semantic typing instead of dynamic enforcement cont'd

While Fennel and Thiemann propose a gradual type system, we can prove that the program is semantically well-typed.

Given $\mathit{addPFB} \triangleq \mathit{addPrivileged}$ false $\mathit{sendToFacebook}$ then

$$\cdot \mid \cdot \mid \cdot \vDash \mathit{addPFB} \approx_{\zeta} \mathit{addPFB} : \mathsf{ref}(\mathit{Report}^{\perp}) \xrightarrow{\perp} 1$$

Value-dependent classification

Consider

$$\begin{aligned} \textit{valDep} \triangleq \lambda f.\, & \mathsf{let}\, d = \mathsf{ref}(\mathsf{true}, \mathit{secret})\, \mathsf{in} \\ & f\ d; \\ & \mathsf{let}\, (b, v) = !\, d\, \mathsf{in} \\ & \mathsf{if}\, b\, \mathsf{then}\, 42\, \mathsf{else}\, v \end{aligned}$$

Ideally, $[\![\mathbb{N}^{\perp}]\!](\textit{valDep}\ f, \textit{valDep}\ f)$, but only for f that maintain the invariant

$$\exists b, v_L, v_R. d_L \mapsto_L (b, v_L) * d_R \mapsto_R (b, v_R) * [\![\mathbb{N}^{\mathsf{if}\ b\ \mathsf{then}\ \top\ \mathsf{else}\ \perp}]\!](v_L, v_R)$$

Value-dependent classification cont'd

However, this burdens the client with proof obligations. Instead, we can exploit existential packs to conceal the proof obligations. E.g.,

$$
\begin{aligned}
\textit{valDepPack} \triangleq\ & \mathsf{let}\ get = \lambda\, d.\ \mathsf{let}\ (b, v) = !\, d\ \mathsf{in}\ \mathsf{if}\ b\ \mathsf{then}\ \mathsf{inj}_1\ v\ \mathsf{else}\ \mathsf{inj}_2\ v\ \mathsf{in} \\
& \mathsf{let}\ setL = \lambda\, d, v.\ d \leftarrow (\mathsf{false}, v)\ \mathsf{in} \\
& \mathsf{let}\ setH = \lambda\, d, v.\ d \leftarrow (\mathsf{true}, v)\ \mathsf{in} \\
& \mathsf{pack}\, (\mathsf{ref}(\mathsf{true}, secret), get, setL, setH)
\end{aligned}
$$

for which it holds

$$
\begin{aligned}
\cdot \mid \cdot \mid \cdot \vDash \textit{valDepPack} \approx_{\zeta} \textit{valDepPack} :& \\
\exists \alpha.\ \left(\alpha^{\bot} \times \left(\alpha^{\bot} \xrightarrow{\top} \mathbb{N}^{\top} + \mathbb{N}^{\bot}\right) \times \left(\alpha^{\bot} \xrightarrow{\top} \mathbb{N}^{\bot} \xrightarrow{\bot} 1\right) \times \left(\alpha^{\bot} \xrightarrow{\top} \mathbb{N}^{\top} \xrightarrow{\bot} 1\right)\right)&
\end{aligned}
$$

Conclusion

In summary, we have

- defined a novel semantic model of an expressive IFC type system with support for impredicative polymorphism, label polymorphism, recursive types, and general references,
  - unary and binary logical-relations models
  - a theory of Modal Weakest Preconditions
- showed that the type system entails termination-insensitive noninterference, and
- illustrated how the model can be used to reason about syntactically ill-typed but semantically secure code with compositional integration.

Conclusion

In summary, we have

- defined a novel semantic model of an expressive IFC type system with support for impredicative polymorphism, label polymorphism, recursive types, and general references,
    - unary and binary logical-relations models
    - a theory of Modal Weakest Preconditions
- showed that the type system entails termination-insensitive noninterference, and
- illustrated how the model can be used to reason about syntactically ill-typed but semantically secure code with compositional integration.

So, what's next?

With our model, we believe to have a very strong methodology for establishing TINI.

- Other security notions (termination-sensitive, progress-sensitive, ...)[2]
- Security libraries (LIO, MAC, ...)
- Concurrency
- Declassification
- ???

---

[2]See Frumin et al. (S&P '21) for an approach in Iris.

Thank you

Extra slides

Unary-binary MWP lemma

Lemma (Unary-binary step-taking update MWPs)

$$\mathsf{mwp}_{\mathcal{E}}^{\mathcal{M}\Rrightarrow\triangleright}\, e_1 \left\{ v.\ \mathsf{mwp}_{\mathcal{E}}^{\mathcal{M}\Rrightarrow\triangleright}\, e_2 \left\{ w.\ \Phi(v, w) \right\} \right\} \twoheadrightarrow\!\!\ast\ \mathsf{mwp}\, e_1 \sim e_2 \left\{ \Phi \right\}$$

$$\mathsf{mwp}_{\mathcal{E}}^{\mathcal{M}\Rrightarrow\triangleright}\, e_2 \left\{ w.\ \mathsf{mwp}_{\mathcal{E}}^{\mathcal{M}\Rrightarrow\triangleright}\, e_1 \left\{ v.\ \Phi(v, w) \right\} \right\} \twoheadrightarrow\!\!\ast\ \mathsf{mwp}\, e_1 \sim e_2 \left\{ \Phi \right\}$$

Why the binary-unary subsumption property?

Let's prove the compatibility lemma for conditional expressions:

Lemma

$$\Xi \mid \Psi \mid \Gamma \vDash \text{if } e \text{ then } e_1 \text{ else } e_2 \approx_\zeta \text{if } e' \text{ then } e'_1 \text{ else } e'_2 : \tau$$

given well-typed sub-terms and $\Xi \mid \Psi \mid \Gamma \vDash e \approx_\zeta e' : \mathbb{B}^\ell$, $\Xi \mid \Psi \mid \Gamma \vDash e_i \approx_\zeta e'_i : \tau$, *and* $\tau \searrow \ell$.

Why the binary-unary subsumption property?

Let's prove the compatibility lemma for conditional expressions:

Lemma

$$\Xi \mid \Psi \mid \Gamma \vDash \mathsf{if}\ e\ \mathsf{then}\ e_1\ \mathsf{else}\ e_2 \approx_\zeta \mathsf{if}\ e'\ \mathsf{then}\ e'_1\ \mathsf{else}\ e'_2 : \tau$$

given well-typed sub-terms and $\Xi \mid \Psi \mid \Gamma \vDash e \approx_\zeta e' : \mathbb{B}^\ell$*,* $\Xi \mid \Psi \mid \Gamma \vDash e_i \approx_\zeta e'_i : \tau$*, and* $\tau \searrow \ell$*.*

Proof.

Unfolding the definition of the judgment, we have to show

$$\mathcal{E}[\![\tau]\!]^\rho_\Theta(\mathsf{if}\ e[\vec{v}/\vec{x}]\ \mathsf{then}\ e_1[\vec{v}/\vec{x}]\ \mathsf{else}\ e_2[\vec{v}/\vec{x}], \mathsf{if}\ e'[\vec{v'}/\vec{x}]\ \mathsf{then}\ e'_1[\vec{v'}/\vec{x}]\ \mathsf{else}\ e'_2[\vec{v'}/\vec{x}]).$$

given $\mathcal{G}[\![\Gamma]\!]^\rho_\Theta(\vec{v}, \vec{v'})$ and $Coh(\Theta)$.

Why the binary-unary subsumption property? cont'd

The proof continues by considering the label ℓ of the guard:

- if $[\![\ell]\!]_\rho \sqsubseteq \zeta$
- if $[\![\ell]\!]_\rho \not\sqsubseteq \zeta$

Why the binary-unary subsumption property? cont'd

The proof continues by considering the label ℓ of the guard:

- if $[\![\ell]\!]_\rho \sqsubseteq \zeta$
  $\Rightarrow\ e \Downarrow v$ and $e' \Downarrow v'$ such that $[\![\mathbb{B}]\!]^\rho_\Theta(v, v')$ meaning $v = v'$.

- if $[\![\ell]\!]_\rho \not\sqsubseteq \zeta$

Why the binary-unary subsumption property? cont'd

The proof continues by considering the label ℓ of the guard:

- if $[\![\ell]\!]_\rho \sqsubseteq \zeta$

  $\Rightarrow$ $e \Downarrow v$ and $e' \Downarrow v'$ such that $[\![\mathbb{B}]\!]^\rho_\Theta(v, v')$ meaning $v = v'$.

  $\Rightarrow$ We have to show $\mathcal{E}[\![\tau]\!]^\rho_\Theta(e_1[\vec{v}/\vec{x}], e_1'[\vec{v}/\vec{x}])$ and $\mathcal{E}[\![\tau]\!]^\rho_\Theta(e_1[\vec{v}/\vec{x}], e_1'[\vec{v}/\vec{x}])$. $\checkmark$
- if $[\![\ell]\!]_\rho \not\sqsubseteq \zeta$

Why the binary-unary subsumption property? cont'd

The proof continues by considering the label ℓ of the guard:

- if $[\![\ell]\!]_\rho \sqsubseteq \zeta$
  - $\Rightarrow$ $e \Downarrow v$ and $e' \Downarrow v'$ such that $[\![\mathbb{B}]\!]_\Theta^\rho(v, v')$ meaning $v = v'$.
  - $\Rightarrow$ We have to show $\mathcal{E}[\![\tau]\!]_\Theta^\rho(e_1[\vec{v}/\vec{x}], e_1'[\vec{v}/\vec{x}])$ and $\mathcal{E}[\![\tau]\!]_\Theta^\rho(e_1[\vec{v}/\vec{x}], e_1'[\vec{v}/\vec{x}])$. ✓
- if $[\![\ell]\!]_\rho \not\sqsubseteq \zeta$
  - $\Rightarrow$ $e \Downarrow v$ and $e' \Downarrow v'$ such that $[\![\mathbb{B}]\!]_\Theta^\rho(v)$ and $[\![\mathbb{B}]\!]_\Theta^\rho(v')$

Why the binary-unary subsumption property? cont'd

The proof continues by considering the label ℓ of the guard:

- if $[\![\ell]\!]_\rho \sqsubseteq \zeta$

  $\Rightarrow$ $e \Downarrow v$ and $e' \Downarrow v'$ such that $[\![\mathbb{B}]\!]^\rho_\Theta(v, v')$ meaning $v = v'$.

  $\Rightarrow$ We have to show $\mathcal{E}[\![\tau]\!]^\rho_\Theta(e_1[\vec{v}/\vec{x}], e'_1[\vec{v}/\vec{x}])$ and $\mathcal{E}[\![\tau]\!]^\rho_\Theta(e_1[\vec{v}/\vec{x}], e'_1[\vec{v}/\vec{x}])$. $\checkmark$

- if $[\![\ell]\!]_\rho \not\sqsubseteq \zeta$

  $\Rightarrow$ $e \Downarrow v$ and $e' \Downarrow v'$ such that $[\![\mathbb{B}]\!]^\rho_\Theta(v)$ and $[\![\mathbb{B}]\!]^\rho_\Theta(v')$

  $\Rightarrow$ We have 4 cases

  - $\mathcal{E}[\![\tau]\!]^\rho_\Theta(e_1[\vec{v}/\vec{x}], e'_1[\vec{v}/\vec{x}])$ $\checkmark$
  - $\mathcal{E}[\![\tau]\!]^\rho_\Theta(e_2[\vec{v}/\vec{x}], e'_2[\vec{v}/\vec{x}])$ $\checkmark$
  - $\mathcal{E}[\![\tau]\!]^\rho_\Theta(e_1[\vec{v}/\vec{x}], e'_2[\vec{v}/\vec{x}])$
  - $\mathcal{E}[\![\tau]\!]^\rho_\Theta(e_2[\vec{v}/\vec{x}], e'_1[\vec{v}/\vec{x}])$

Why the binary-unary subsumption property? cont'd cont'd

Given $\tau = t^{\ell'}$ as $[\![\ell]\!]_\rho \not\sqsubseteq \zeta$ and $\tau \searrow \ell$ then $[\![\ell']\!]_\rho \not\sqsubseteq \zeta$.

Hence

$$\begin{aligned}
\mathcal{E}[\![\tau]\!]^\rho_\Theta(e_1[\vec{v}/\vec{x}], e_2'[\vec{v}/\vec{x}]) &= \mathsf{mwp}\, e_1[\vec{v}/\vec{x}] \sim e_2'[\vec{v}/\vec{x}]\, \{v, v'.\, [\![\tau]\!]^\rho_\Theta(v, v')\} \\
&= \mathsf{mwp}\, e_1[\vec{v}/\vec{x}] \sim e_2'[\vec{v}/\vec{x}]\, \{v, v'.\, [\![\tau]\!]^\rho_\Theta(v) * [\![\tau]\!]^\rho_\Theta(v')\}
\end{aligned}$$

Why the binary-unary subsumption property? cont'd cont'd

Given $\tau = t^{\ell'}$ as $[\![\ell]\!]_\rho \not\sqsubseteq \zeta$ and $\tau \searrow \ell$ then $[\![\ell']\!]_\rho \not\sqsubseteq \zeta$.

Hence

$$\begin{aligned}\mathcal{E}[\![\tau]\!]^\rho_\Theta(e_1[\vec{v}/\vec{x}], e_2'[\vec{v}/\vec{x}]) &= \text{mwp}\; e_1[\vec{v}/\vec{x}] \sim e_2'[\vec{v}/\vec{x}]\;\{v, v'.\; [\![\tau]\!]^\rho_\Theta(v, v')\} \\ &= \text{mwp}\; e_1[\vec{v}/\vec{x}] \sim e_2'[\vec{v}/\vec{x}]\;\{v, v'.\; [\![\tau]\!]^\rho_\Theta(v) * [\![\tau]\!]^\rho_\Theta(v')\}\end{aligned}$$

With the unary-binary MWP lemma and the fundamental theorem, we should be done.

Theorem (Unary fundamental theorem)

If $\quad \Xi \mid \Psi \mid \Gamma \vdash_{pc} e : \tau \quad$ *then* $\quad \Xi \mid \Psi \mid \Gamma \vDash_{pc} e : \tau.$

However, we need $\mathcal{G}[\![\Gamma]\!]^\rho_\Theta(\vec{v})$ and $\mathcal{G}[\![\Gamma]\!]^\rho_\Theta(\vec{v'})$—which follows from subsumption!

Type system cont'd

T-TLAM

$$\frac{\Xi, \alpha \mid \Psi \mid \Gamma \vdash_{\ell_e} e : \tau}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \Lambda\, e : (\forall_{\ell_e}\, \alpha.\, \tau)^{\perp}}$$

T-LLAM

$$\frac{\Xi \mid \Psi, \kappa \mid \Gamma \vdash_{\ell_e} e : \tau \qquad \mathsf{FV}(\ell_e) \subseteq \Psi \cup \{\kappa\}}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \Lambda\!\!\Lambda\, e : (\forall\!\!\forall_{\ell_e}\, \kappa.\, \tau)^{\perp}}$$

T-TAPP

$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : (\forall_{\ell_e}\, \alpha.\, \tau)^{\ell} \qquad \Psi \vdash pc \sqcup \ell \sqsubseteq \ell_e \qquad \mathsf{FV}(t) \subseteq \Xi}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e \; \_ : \tau[t/\alpha]}$$

T-LAPP

$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : (\forall\!\!\forall_{\ell_e}\, \kappa.\, \tau)^{\ell} \qquad \Psi \vdash pc \sqcup \ell \sqsubseteq \ell_e[\ell'/\kappa] \qquad \Psi \vdash \tau[\ell'/\kappa] \searrow \ell \qquad \mathsf{FV}(\ell') \subseteq \Psi}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e \; \_ : \tau[\ell'/\kappa]}$$

Type system cont'd

T-TLAM

$$\frac{\Xi, \alpha \mid \Psi \mid \Gamma \vdash_{\ell_e} e : \tau}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \Lambda\, e : \left(\forall_{\ell_e}\, \alpha.\, \tau\right)^{\perp}}$$

T-LLAM

$$\frac{\Xi \mid \Psi, \kappa \mid \Gamma \vdash_{\ell_e} e : \tau \qquad \mathsf{FV}(\ell_e) \subseteq \Psi \cup \{\kappa\}}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \Lambda\!\!\!\Lambda\, e : \left(\forall\!\!\!\forall_{\ell_e}\, \kappa.\, \tau\right)^{\perp}}$$

T-TAPP

$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : \left(\forall_{\ell_e}\, \alpha.\, \tau\right)^{\ell} \qquad \Psi \vdash pc \sqcup \ell \sqsubseteq \ell_e \qquad \mathsf{FV}(t) \subseteq \Xi}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e\ \_ : \tau[t/\alpha]}$$

T-LAPP

$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : \left(\forall\!\!\!\forall_{\ell_e}\, \kappa.\, \tau\right)^{\ell} \qquad \Psi \vdash pc \sqcup \ell \sqsubseteq \ell_e[\ell'/\kappa] \qquad \Psi \vdash \tau[\ell'/\kappa] \searrow \ell \qquad \mathsf{FV}(\ell') \subseteq \Psi}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e\ \_ : \tau[\ell'/\kappa]}$$

Type system cont'd

T-TLAM

$$\frac{\Xi, \alpha \mid \Psi \mid \Gamma \vdash_{\ell_e} e : \tau}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \Lambda\, e : (\forall_{\ell_e} \alpha.\, \tau)^{\perp}}$$

T-LLAM

$$\frac{\Xi \mid \Psi, \kappa \mid \Gamma \vdash_{\ell_e} e : \tau \qquad \mathsf{FV}(\ell_e) \subseteq \Psi \cup \{\kappa\}}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \mathbb{\Lambda}\, e : (\mathbb{\forall}_{\ell_e} \kappa.\, \tau)^{\perp}}$$

T-TAPP

$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : (\forall_{\ell_e} \alpha.\, \tau)^{\ell} \qquad \Psi \vdash pc \sqcup \ell \sqsubseteq \ell_e \qquad \mathsf{FV}(t) \subseteq \Xi}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e\ \_ : \tau[t/\alpha]}$$

T-LAPP

$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : (\forall_{\ell_e} \kappa.\, \tau)^{\ell} \qquad \Psi \vdash pc \sqcup \ell \sqsubseteq \ell_e[\ell'/\kappa] \qquad \Psi \vdash \tau[\ell'/\kappa] \searrow \ell \qquad \mathsf{FV}(\ell') \subseteq \Psi}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e\ \_ : \tau[\ell'/\kappa]}$$

What we want to do cont'd

Intuitively, this is done by defining

$$e_1 \approx e_2 : \tau \triangleq e_1 \to^* v_1 \wedge e_2 \to^* v_2 \Rightarrow [\![\tau]\!](v_1, v_2)$$

However, as we have references, we hit the **type-world circularity** problem:

$$[\![\mathsf{ref}(\tau)]\!](W) = \{\iota \mid \iota \in \mathrm{dom}(W) \wedge W(\iota) = [\![\tau]\!]\}$$

implies

$$\begin{aligned} [\![\tau]\!] &: T \\ T &= \mathit{World} \to \mathit{Pred}(\mathit{Val}) \\ \mathit{World} &= \mathit{Loc} \to T \end{aligned}$$

but this domain does not exist …

What we want to do cont'd

Intuitively, this is done by defining

$$e_1 \approx e_2 : \tau \triangleq e_1 \to^* v_1 \land e_2 \to^* v_2 \Rightarrow [\![\tau]\!](v_1, v_2)$$

However, as we have references, we hit the **type-world circularity** problem:

$$[\![\mathsf{ref}(\tau)]\!](W) = \{\iota \mid \iota \in \mathrm{dom}(W) \land W(\iota) = [\![\tau]\!]\}$$

implies

$$[\![\tau]\!] : T$$

$$T = \mathit{World} \to \mathit{Pred}(\mathit{Val})$$

$$\mathit{World} = \mathit{Loc} \to T$$

but this domain does not exist ...

What we want to do cont'd

Intuitively, this is done by defining

$$e_1 \approx e_2 : \tau \triangleq e_1 \to^* v_1 \wedge e_2 \to^* v_2 \Rightarrow [\![\tau]\!](v_1, v_2)$$

However, as we have references, we hit the **type-world circularity** problem:

$$[\![\mathsf{ref}(\tau)]\!](W) = \{\iota \mid \iota \in \mathrm{dom}(W) \wedge W(\iota) = [\![\tau]\!]\}$$

implies

$$\begin{aligned} [\![\tau]\!] &: T \\ T &= \mathit{World} \to \mathit{Pred}(\mathit{Val}) \\ \mathit{World} &= \mathit{Loc} \to T \end{aligned}$$

but this domain does not exist …

MWP instances

Example (Unary step-taking update modality)

$$\mathsf{mwp}_{\mathcal{E}}^{\mathcal{M}_{\Rrightarrow\triangleright}}\, e\,\{\Phi\} = \forall \sigma, \sigma', v, n.\, (e, \sigma) \rightarrow^{n} (v, \sigma') \mathrel{-\!\!*} S(\sigma) \mathrel{-\!\!*} \left({}^{\mathcal{E}}\!\Rrightarrow^{\emptyset} \triangleright {}^{\emptyset}\!\Rrightarrow^{\mathcal{E}}\right)^{n} \Rrightarrow_{\mathcal{E}} (\Phi(v) * S(\sigma')) .$$

Example (Binary step-taking update modality)

$$\mathsf{mwp}_{\mathcal{E}}\, e_1 \sim e_2\,\{\Phi\} = \forall \sigma_1, \sigma_1', v, n.\, (e_1, \sigma_1) \rightarrow^{n} (v, \sigma_1') \mathrel{-\!\!*} S_1(\sigma_1) \mathrel{-\!\!*} \forall \sigma_2, \sigma_2', w, m.\, (e_2, \sigma_2) \rightarrow^{m} (w, \sigma_2') \mathrel{-\!\!*} S_2(\sigma_2) \mathrel{-\!\!*} \left({}^{\mathcal{E}}\!\Rrightarrow^{\emptyset} \triangleright {}^{\emptyset}\!\Rrightarrow^{\mathcal{E}}\right)^{n+m} \Rrightarrow_{\mathcal{E}} (\Phi(v, w) * S_1(\sigma_1') * S_2(\sigma_2'))$$

Semantic typing judgment

Our semantic typing judgment now follows:

$$\Xi \mid \Psi \mid \Gamma \vDash e \approx_\zeta e' : \tau \triangleq \Box \left(\begin{array}{r} \forall \Theta, \rho, \vec{v}, \vec{v'}.\, \mathrm{dom}(\Xi) \subseteq \mathrm{dom}(\Theta) * \mathrm{dom}(\Psi) \subseteq \mathrm{dom}(\rho) \twoheadrightarrow \\ \mathit{Coh}(\Theta) * \mathcal{G}[\![\Gamma]\!]^\rho_\Theta(\vec{v}, \vec{v'}) \twoheadrightarrow \mathcal{E}[\![\tau]\!]^\rho_\Theta(e[\vec{v}/\vec{x}], e'[\vec{v'}/\vec{x}]) \end{array} \right)$$

given

$$\begin{aligned} \mathcal{G}[\![\cdot]\!]^\rho_\Theta(\epsilon, \epsilon) &\triangleq \textsf{True} \\ \mathcal{G}[\![\Gamma, x : \tau]\!]^\rho_\Theta(\vec{v}w, \vec{v'}w') &\triangleq \mathcal{G}[\![\Gamma]\!]^\rho_\Theta(\vec{v}, \vec{v'}) * [\![\tau]\!]^\rho_\Theta(w, w') \end{aligned}$$

Unary value relation

$$\llbracket t^\ell \rrbracket^\rho_\Delta(v) \triangleq \llbracket t \rrbracket^\rho_\Delta(v)$$

$$\llbracket \tau_1 \xrightarrow{\ell_e} \tau_2 \rrbracket^\rho_\Delta(v) \triangleq \square\,(\forall w.\, \llbracket \tau_1 \rrbracket^\rho_\Delta(w) \twoheadrightarrow \mathcal{E}_{\ell_e} \llbracket \tau_2 \rrbracket^\rho_\Delta(v\ w))$$

$$\llbracket \mathsf{ref}(t^\ell) \rrbracket^\rho_\Delta(v) \triangleq \exists \iota, \mathcal{N}.\, v = \iota * \mathcal{R}(\Delta, \rho, \iota, \ell, \mathcal{N})$$

where $\mathcal{R}(\Delta, \rho, \iota, \ell, \mathcal{N})$ is defined by cases:

- if $\llbracket \ell \rrbracket_\rho \sqsubseteq \zeta$ then

$$\square \forall \mathcal{E}.\, \mathcal{N} \subseteq \mathcal{E} \Rightarrow \left({}^{\mathcal{E}}\!\Rrightarrow^{\mathcal{E}\setminus\mathcal{N}} \triangleright \begin{pmatrix} \exists w.\, \iota \mapsto_i w * \llbracket \tau \rrbracket^\rho_\Delta(w) * \\ \left((\triangleright\, \iota \mapsto_i w * \llbracket \tau \rrbracket^\rho_\Delta(w)) \;{}^{\mathcal{E}\setminus\mathcal{N}}\!\!\equiv\!\!*^{\mathcal{E}}\; \mathsf{True} \right) \end{pmatrix} \right)$$

- if $\llbracket \ell \rrbracket_\rho \not\sqsubseteq \zeta$ then

$$\square \forall \mathcal{E}.\, \mathcal{N} \subseteq \mathcal{E} \Rightarrow \left({}^{\mathcal{E}}\!\Rrightarrow^{\mathcal{E}\setminus\mathcal{N}} \triangleright \begin{pmatrix} \exists w.\, \iota \mapsto_i w * \llbracket \tau \rrbracket^\rho_\Delta(w) * \\ \left((\triangleright \exists w'.\, \iota \mapsto_i w' * \llbracket \tau \rrbracket^\rho_\Delta(w')) \;{}^{\mathcal{E}\setminus\mathcal{N}}\!\!\equiv\!\!*^{\mathcal{E}}\; \mathsf{True} \right) \end{pmatrix} \right)$$

Unary value relation

$$\llbracket t^{\ell} \rrbracket^{\rho}_{\Delta}(v) \triangleq \llbracket t \rrbracket^{\rho}_{\Delta}(v)$$

$$\llbracket \tau_1 \xrightarrow{\ell_e} \tau_2 \rrbracket^{\rho}_{\Delta}(v) \triangleq \Box\,(\forall w.\, \llbracket \tau_1 \rrbracket^{\rho}_{\Delta}(w) \mathrel{-\!\!*} \mathcal{E}_{\ell_e} \llbracket \tau_2 \rrbracket^{\rho}_{\Delta}(v\ w))$$

$$\llbracket \mathsf{ref}(t^{\ell}) \rrbracket^{\rho}_{\Delta}(v) \triangleq \exists \iota, \mathcal{N}.\, v = \iota * \mathcal{R}(\Delta, \rho, \iota, \ell, \mathcal{N})$$

where $\mathcal{R}(\Delta, \rho, \iota, \ell, \mathcal{N})$ is defined by cases:

- if $\llbracket \ell \rrbracket_{\rho} \sqsubseteq \zeta$ then

$$\Box \forall \mathcal{E}.\, \mathcal{N} \subseteq \mathcal{E} \Rightarrow \left({}^{\mathcal{E}}{\Rrightarrow}^{\mathcal{E}\setminus\mathcal{N}} \triangleright \begin{pmatrix} \exists w.\, \iota \mapsto_i w * \llbracket \tau \rrbracket^{\rho}_{\Delta}(w) * \\ \left((\triangleright\, \iota \mapsto_i w * \llbracket \tau \rrbracket^{\rho}_{\Delta}(w)) \;{}^{\mathcal{E}\setminus\mathcal{N}}{\Rrightarrow\!\!*}^{\mathcal{E}}\ \mathsf{True} \right) \end{pmatrix} \right)$$

- if $\llbracket \ell \rrbracket_{\rho} \not\sqsubseteq \zeta$ then

$$\Box \forall \mathcal{E}.\, \mathcal{N} \subseteq \mathcal{E} \Rightarrow \left({}^{\mathcal{E}}{\Rrightarrow}^{\mathcal{E}\setminus\mathcal{N}} \triangleright \begin{pmatrix} \exists w.\, \iota \mapsto_i w * \llbracket \tau \rrbracket^{\rho}_{\Delta}(w) * \\ \left((\triangleright \exists w'.\, \iota \mapsto_i w' * \llbracket \tau \rrbracket^{\rho}_{\Delta}(w')) \;{}^{\mathcal{E}\setminus\mathcal{N}}{\Rrightarrow\!\!*}^{\mathcal{E}}\ \mathsf{True} \right) \end{pmatrix} \right)$$

Unary value relation

$$\llbracket t^{\ell} \rrbracket^{\rho}_{\Delta}(v) \triangleq \llbracket t \rrbracket^{\rho}_{\Delta}(v)$$

$$\llbracket \tau_1 \xrightarrow{\ell_e} \tau_2 \rrbracket^{\rho}_{\Delta}(v) \triangleq \Box \left(\forall w.\, \llbracket \tau_1 \rrbracket^{\rho}_{\Delta}(w) \mathrel{-\!\!*} \mathcal{E}_{\ell_e} \llbracket \tau_2 \rrbracket^{\rho}_{\Delta}(v\ w)\right)$$

$$\llbracket \mathsf{ref}(t^{\ell}) \rrbracket^{\rho}_{\Delta}(v) \triangleq \exists \iota, \mathcal{N}.\, v = \iota * \mathcal{R}(\Delta, \rho, \iota, \ell, \mathcal{N})$$

where $\mathcal{R}(\Delta, \rho, \iota, \ell, \mathcal{N})$ is defined by cases:

- if $\llbracket \ell \rrbracket_{\rho} \sqsubseteq \zeta$ then

$$\Box \forall \mathcal{E}.\, \mathcal{N} \subseteq \mathcal{E} \Rightarrow \left({}^{\mathcal{E}}\!\!\Rrightarrow^{\mathcal{E}\setminus\mathcal{N}} \triangleright \left(\begin{array}{l} \exists w.\, \iota \mapsto_i w * \llbracket \tau \rrbracket^{\rho}_{\Delta}(w) * \\ \left((\triangleright\, \iota \mapsto_i w * \llbracket \tau \rrbracket^{\rho}_{\Delta}(w)) \ {}^{\mathcal{E}\setminus\mathcal{N}}\!\!\Rrightarrow\!\!*^{\mathcal{E}}\ \mathsf{True} \right) \end{array} \right) \right)$$

- if $\llbracket \ell \rrbracket_{\rho} \not\sqsubseteq \zeta$ then

$$\Box \forall \mathcal{E}.\, \mathcal{N} \subseteq \mathcal{E} \Rightarrow \left({}^{\mathcal{E}}\!\!\Rrightarrow^{\mathcal{E}\setminus\mathcal{N}} \triangleright \left(\begin{array}{l} \exists w.\, \iota \mapsto_i w * \llbracket \tau \rrbracket^{\rho}_{\Delta}(w) * \\ \left((\triangleright \exists w'.\, \iota \mapsto_i w' * \llbracket \tau \rrbracket^{\rho}_{\Delta}(w')) \ {}^{\mathcal{E}\setminus\mathcal{N}}\!\!\Rrightarrow\!\!*^{\mathcal{E}}\ \mathsf{True} \right) \end{array} \right) \right)$$

Computing with memoization

Consider the following memoization utility

$$
\begin{aligned}
&\textit{memoize} \triangleq \lambda f, init. \\
&\qquad\qquad \mathsf{let}\ cache = \mathsf{ref}(init, f\ \ init)\ \mathsf{in} \\
&\qquad\qquad \mathsf{let}\ recompute = \lambda v.\ \mathsf{let}\ result = f\ \ v\ \mathsf{in}\ cache \leftarrow (v, result); result\ \mathsf{in} \\
&\qquad\qquad \lambda v.\ \mathsf{let}\ (w, result) = !\ cache\ \mathsf{in} \\
&\qquad\qquad\quad \mathsf{if}\ v = w\ \mathsf{then}\ result\ \mathsf{else}\ recompute\ \ v
\end{aligned}
$$

We would like that, e.g., for $f : \mathbb{N}^\ell \xrightarrow{\top} \mathbb{N}^\ell$ then *memoize* $f\ 0$ is interchangeable with f. However, we cannot statically type *memoize*.

Computing with memoization cont'd

Moreover, f needs to satisfy a semantic condition; if not, consider e.g.

$$\begin{aligned}
&\mathsf{let}\ counter = \mathsf{ref}(0)\ \mathsf{in} \\
&\mathsf{let}\ f' = \textbf{\textit{memoize}}\ (\lambda_.\ counter \leftarrow (!\,counter + 1); !\,counter)\ 0\ \mathsf{in} \\
&\mathsf{if}\ secret\ \mathsf{then}\ f'\ 0\ \mathsf{else}\ (); \\
&f'\ 0
\end{aligned}$$

However, for any "purely acting" function f, we have that

$$\cdot \mid \cdot \mid \cdot \vDash \textbf{\textit{memoize}}\ f\ 0 \approx_\zeta \textbf{\textit{memoize}}\ f\ 0 : \mathbb{N}^\ell \xrightarrow{\top} \mathbb{N}^\ell$$