Mechanized Logical Relations for Termination-Insensitive Noninterference

Technical Appendix

August 6, 2020

Abstract

This document presents a λ_{sec}, a standard ML-like language with higher-order heap equipped with an information-flow control type system featuring subtyping, recursive types, label polymorphism, existential types, and impredicative type polymorphism. We introduce a generalized theory of Modal Weakest Precondition predicates and construct a novel "'logical" logical-relations model of the type system in Iris, a state-of-the-art separation logic. Finally, we use the model to prove that the type system guarantees termination-insensitive noninterference.

1 Syntax and Semantics

Definition 1.1 (Syntax and types).

$$
\begin{array}{rcl}
x, y, z & \in & \mathit{Var} \\
\iota & \in & \mathit{Loc} \\
n & \in & \mathbb{N} \\
l, \zeta & \in & \mathcal{L} \\
\circledcirc & ::= & + \mid - \mid * \mid = \mid < \\
\ell, pc \in \mathit{Label}_{\mathcal{L}} & ::= & \kappa \mid l \mid \ell \sqcup \ell \\
\tau \in \mathit{LType} & ::= & t^{\ell} \\
t \in \mathit{Type} & ::= & \alpha \mid 1 \mid \mathbb{B} \mid \mathbb{N} \mid \tau \times \tau \mid \tau + \tau \mid \tau \xrightarrow{\ell} \tau \mid \forall_{\ell}\, \alpha.\, \tau \mid \forall\!\!\!\forall_{\ell}\, \kappa.\, \tau \mid \exists \alpha.\, \tau \mid \mu\alpha.\, \tau \mid \mathsf{ref}(\tau) \\
e \in \mathit{Expr} & ::= & x \mid () \mid \mathsf{true} \mid \mathsf{false} \mid n \mid n \circledcirc n \mid \lambda x.\, e \mid e\; e \mid \Lambda\, e \mid \Lambda\!\!\!\Lambda\, e \mid e\; \_ \mid \\
& \mid & \mathsf{if}\; e\; \mathsf{then}\; e\; \mathsf{else}\; e \mid (e, e) \mid \pi_i\, e \mid \mathsf{inj}_i\, e \mid \mathsf{match}\; e\; \mathsf{with}\; \mathsf{inj}_i \Rightarrow e_i\; \mathsf{end} \\
& \mid & \mathsf{ref}(e) \mid\, !\, e \mid e \leftarrow e \mid \mathsf{fold}\;\; e \mid \mathsf{unfold}\;\; e \mid \mathsf{pack}\, e \mid \mathsf{unpack}\; e\; \mathsf{as}\; x\; \mathsf{in}\; e \\
v \in \mathit{Val} & ::= & () \mid \mathsf{true} \mid \mathsf{false} \mid n \mid \lambda x.\, e \mid \Lambda\, e \mid \Lambda\!\!\!\Lambda\, e \mid \mathsf{fold}\;\; v \mid \mathsf{pack}\, v \mid (v, v) \mid \mathsf{inj}_i\, v \mid \iota \\
K \in \mathit{ECtx} & ::= & - \mid K \circledcirc e \mid v \circledcirc K \mid \mathsf{if}\; K\; \mathsf{then}\; e\; \mathsf{else}\; e \mid (K, e) \mid (v, K) \mid \pi_1\, K \mid \pi_2\, K \\
& \mid & \mathsf{inj}_1\, K \mid \mathsf{inj}_2\, K \mid \mathsf{match}\; K\; \mathsf{with}\; \mathsf{inj}_i \Rightarrow e_i\; \mathsf{end} \mid K\; e \mid v\; K \\
& \mid & \mathsf{ref}(K) \mid\, !\, K \mid K \leftarrow e \mid v \leftarrow K \mid \mathsf{fold}\, K \mid \mathsf{unfold}\, K \mid \mathsf{pack}\, K \mid \mathsf{unpack}\; K\; \mathsf{as}\; x\; \mathsf{in}\; e \\
\sigma & \in & \mathit{Loc} \xrightarrow{\text{fin}} \mathit{Val}
\end{array}
$$

In addition to the given constructions we will write $\mathsf{let}\; x = e_1\; \mathsf{in}\; e_2$ for the term $(\lambda x.\, e_1)\; e_2$ and $e_1; e_2$ for $\mathsf{let}\; \_ = e_1\; \mathsf{in}\; e_2$.

The syntax of types is parameterized over a bounded join-semilattice $\mathcal{L}$ where the induced ordering $\sqsubseteq$ defines the security policy. $\forall\!\!\!\forall_{\ell}\, \kappa.\, \tau$ denotes the type of label-polymorphic terms (over variable κ) with the corresponding term $\Lambda\!\!\!\Lambda\, e$. $\forall_{\ell}\, \alpha.\, \tau$ denotes the type of type-polymorphic terms (over variable α) with the corresponding term $\Lambda\, e$. Both the two polymorphic types and the arrow type are annotated with a label ℓ that in the type system will constitute a lower-bound on side-effects of the term.

Definition 1.2 (Operational semantics).

$$
\begin{array}{rcll}
v \circledcirc v' & \overset{\text{pure}}{\rightsquigarrow} & v'' & \text{if } v'' = v \circledcirc v' \\
\textsf{if true then } e_1 \textsf{ else } e_2 & \overset{\text{pure}}{\rightsquigarrow} & e_1 & \\
\textsf{if false then } e_1 \textsf{ else } e_2 & \overset{\text{pure}}{\rightsquigarrow} & e_2 & \\
\pi_i\,(v_1, v_2) & \overset{\text{pure}}{\rightsquigarrow} & v_i & i \in \{1,2\} \\
\textsf{match inj}_i\, v \textsf{ with inj}_i \Rightarrow e \textsf{ end} & \overset{\text{pure}}{\rightsquigarrow} & e[v/x] & i \in \{1,2\} \\
(\lambda x.\, e)\, v & \overset{\text{pure}}{\rightsquigarrow} & e[v/x] & \\
(\Lambda\, e)\, \_ & \overset{\text{pure}}{\rightsquigarrow} & e & \\
(\mathbb{\Lambda}\, e)\, \_ & \overset{\text{pure}}{\rightsquigarrow} & e & \\
\textsf{unfold }\ (\textsf{fold }\ v) & \overset{\text{pure}}{\rightsquigarrow} & v & \\
\textsf{unpack}\,(\textsf{pack}\, v) \textsf{ as } x \textsf{ in } e & \overset{\text{pure}}{\rightsquigarrow} & e[v/x] & \\
(\sigma, e) & \rightarrow_{\mathsf{h}} & (\sigma, e') & \text{if } e \overset{\text{pure}}{\rightsquigarrow} e' \\
(\sigma, \textsf{ref}(v)) & \rightarrow_{\mathsf{h}} & (\sigma[\iota \mapsto v], \iota) & \text{if } \iota \notin \text{dom}(\sigma) \\
(\sigma, !\,\iota) & \rightarrow_{\mathsf{h}} & (\sigma, \sigma(\iota)) & \text{if } \iota \in \text{dom}(\sigma) \\
(\sigma, \iota \leftarrow v) & \rightarrow_{\mathsf{h}} & (\sigma[\iota \mapsto v], ()) & \text{if } \iota \in \text{dom}(\sigma)
\end{array}
$$

$$
\frac{(\sigma, e) \rightarrow_{\mathsf{h}} (\sigma', e')}{(\sigma, K[e]) \rightarrow (\sigma', K[e'])}
$$

The operational semantics are mostly standard and defined with a call-by-value, left-to-right evaluation strategy. We first define a head reduction relation, $(\sigma, e) \rightarrow_{\mathsf{h}} (\sigma, e')$, which relates two pairs of a state and an expression. The head-step relation is lifted to a reduction relation $(\sigma, e) \rightarrow (\sigma', e')$ using evaluation contexts.

2 Type System

Definition 2.1 (Label-ordering with free variables).

$$
\textsc{F-refl}\ \frac{\text{FV}(\ell) \subseteq \Psi}{\Psi \vdash \ell \sqsubseteq \ell}
\qquad
\textsc{F-trans}\ \frac{\Psi \vdash \ell_1 \sqsubseteq \ell_2 \qquad \Psi \vdash \ell_2 \sqsubseteq \ell_3}{\Psi \vdash \ell_1 \sqsubseteq \ell_3}
\qquad
\textsc{F-bottom}\ \frac{\text{FV}(\ell) \subseteq \Psi}{\Psi \vdash \bot \sqsubseteq \ell}
\qquad
\textsc{F-label}\ \frac{l_1 \sqsubseteq l_2}{\Psi \vdash l_1 \sqsubseteq l_2}
$$

$$
\textsc{F-join}\ \frac{\Psi \vdash \ell_1 \sqsubseteq \ell_3 \qquad \Psi \vdash \ell_2 \sqsubseteq \ell_3}{\Psi \vdash \ell_1 \sqcup \ell_2 \sqsubseteq \ell_3}
$$

Definition 2.2 (Subtyping).

$$
\textsc{S-refl}\ \frac{\text{FV}(t) \subseteq \Xi}{\Xi \mid \Psi \vdash t <: t}
\qquad
\textsc{S-trans}\ \frac{\Xi \mid \Psi \vdash t_1 <: t_2 \qquad \Xi \mid \Psi \vdash t_2 <: t_3}{\Xi \mid \Psi \vdash t_1 <: t_3}
$$

$$
\textsc{S-arrow}\ \frac{\Xi \mid \Psi \vdash \tau_1' <: \tau_1 \qquad \Xi \mid \Psi \vdash \tau_2 <: \tau_2' \qquad \Psi \vdash \ell_2 \sqsubseteq \ell_1}{\Xi \mid \Psi \vdash \tau_1 \xrightarrow{\ell_1} \tau_2 <: \tau_1' \xrightarrow{\ell_2} \tau_2'}
\qquad
\textsc{S-forall}\ \frac{\Psi \vdash \ell_2 \sqsubseteq \ell_1 \qquad \Xi, \alpha \mid \Psi \vdash \tau_1 <: \tau_2}{\Xi \mid \Psi \vdash \forall_{\ell_1}\, \alpha.\, \tau_1 <: \forall_{\ell_2}\, \alpha.\, \tau_2}
$$

$$
\textsc{S-lforall}\ \frac{\Psi, \kappa \vdash \ell_2 \sqsubseteq \ell_1 \qquad \Xi \mid \Psi, \kappa \vdash \tau_1 <: \tau_2}{\Xi \mid \Psi \vdash \mathbb{\forall}_{\ell_1}\, \kappa.\, \tau_1 <: \mathbb{\forall}_{\ell_2}\, \kappa.\, \tau_2}
\qquad
\textsc{S-prod}\ \frac{\Xi \mid \Psi \vdash \tau_1 <: \tau_1' \qquad \Xi \mid \Psi \vdash \tau_2 <: \tau_2'}{\Xi \mid \Psi \vdash \tau_1 \times \tau_2 <: \tau_1' \times \tau_2'}
$$

$$
\textsc{S-sum}\ \frac{\Xi \mid \Psi \vdash \tau_1 <: \tau_1' \qquad \Xi \mid \Psi \vdash \tau_2 <: \tau_2'}{\Xi \mid \Psi \vdash \tau_1 + \tau_2 <: \tau_1' + \tau_2'}
\qquad
\textsc{S-labeled}\ \frac{\Psi \vdash \ell_1 \sqsubseteq \ell_2 \qquad \Xi \mid \Psi \vdash t_1 <: t_2}{\Xi \mid \Psi \vdash {t_1}^{\ell_1} <: {t_2}^{\ell_2}}
$$

Definition 2.3 (Protected-at).

$$t^{\ell'} \searrow \ell \triangleq \ell \sqsubseteq \ell'$$

Definition 2.4 (Typing).

$$\textsc{T-Var}\ \frac{x : \tau \in \Gamma}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} x : \tau} \qquad \textsc{T-unit}\ \Xi \mid \Psi \mid \Gamma \vdash_{pc} () : 1^{\perp} \qquad \textsc{T-bool}\ \frac{b \in \{\mathsf{true}, \mathsf{false}\}}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} b : \mathbb{B}^{\perp}} \qquad \textsc{T-nat}\ \frac{n \in \mathbb{N}}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} n : \mathbb{N}^{\perp}}$$

$$\textsc{T-binop}\ \frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1 : \mathbb{N}^{\ell_1} \qquad \Xi \mid \Psi \mid \Gamma \vdash_{pc} e_2 : \mathbb{N}^{\ell_2} \qquad \circledcirc : \mathbb{N} \times \mathbb{N} \Rightarrow t}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1 \circledcirc e_2 : t^{\ell_1 \sqcup \ell_2}} \qquad \textsc{T-lam}\ \frac{\Xi \mid \Psi \mid \Gamma, x : \tau_1 \vdash_{\ell_e} e : \tau_2}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \lambda x.\, e : (\tau_1 \xrightarrow{\ell_e} \tau_2)^{\perp}}$$

$$\textsc{T-app}\ \frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1 : (\tau_1 \xrightarrow{\ell_e} \tau_2)^{\ell} \qquad \Xi \mid \Psi \mid \Gamma \vdash_{pc} e_2 : \tau_1 \qquad \Psi \vdash \tau_2 \searrow \ell \qquad \Psi \vdash pc \sqcup \ell \sqsubseteq \ell_e}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1\ e_2 : \tau_2}$$

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

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

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

$$\textsc{T-lapp}\ \frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : (\forall\!\!\forall_{\ell_e}\, \kappa.\, \tau)^{\ell} \qquad \Psi \vdash pc \sqcup \ell \sqsubseteq \ell_e[\ell'/\kappa] \qquad \Psi \vdash \tau[\ell'/\kappa] \searrow \ell \qquad \mathsf{FV}(\ell') \subseteq \Psi}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e\ \_ : \tau[\ell'/\kappa]}$$

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

$$\textsc{T-pair}\ \frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e_1 : \tau_1 \qquad \Xi \mid \Psi \mid \Gamma \vdash_{pc} e_2 : \tau_2}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} (e_1, e_2) : (\tau_1 \times \tau_2)^{\perp}}$$

$$\textsc{T-proj}\ \frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : (\tau_1 \times \tau_2)^{\ell} \qquad \Psi \vdash \tau_i \searrow \ell \qquad i \in \{1, 2\}}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \pi_i\, e : \tau_i} \qquad \textsc{T-inj}\ \frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : \tau_i \qquad i \in \{1, 2\}}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \mathsf{inj}_i\, e : (\tau_1 + \tau_2)^{\perp}}$$

$$\textsc{T-match}\ \frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : (\tau_1 + \tau_2)^{\ell} \qquad \forall i \in \{1, 2\}\,.\, \Xi \mid \Psi \mid \Gamma, x : \tau_i \vdash_{pc \sqcup \ell} e_i : \tau \qquad \Psi \vdash \tau \searrow \ell}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \mathsf{match}\, e\, \mathsf{with}\, \mathsf{inj}_i \Rightarrow e_i\, \mathsf{end} : \tau}$$

$$\textsc{T-fold}\ \frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : \tau[\mu\alpha.\, \tau/\alpha]}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \mathsf{fold}\ e : (\mu\alpha.\, \tau)^{\perp}} \qquad \textsc{T-unfold}\ \frac{\Psi \vdash \tau[\mu\alpha.\, \tau/\alpha] \searrow \ell \qquad \Xi \mid \Psi \mid \Gamma \vdash_{pc} e : (\mu\alpha.\, \tau)^{\ell}}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \mathsf{unfold}\ e : \tau[\mu\alpha.\, \tau/\alpha]}$$

$$\textsc{T-pack}\ \frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : \tau[t/\alpha]}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \mathsf{pack}\, e : (\exists\alpha.\, \tau)^{\perp}}$$

$$\textsc{T-unpack}\ \frac{\Psi \vdash \tau \searrow \ell \qquad \Xi \mid \Psi \mid \Gamma \vdash_{pc} \mathsf{pack}\, e_1 : (\exists\alpha.\, \tau')^{\ell} \qquad \Xi, \alpha \mid \Psi \mid \Gamma, x : \tau' \vdash_{pc \sqcup \ell} e_2 : \tau}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \mathsf{unpack}\, e_1\, \mathsf{as}\, x\, \mathsf{in}\, e_2 : \tau}$$

$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : \tau \qquad \Psi \vdash \tau \searrow pc}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \mathsf{ref}(e) : \mathsf{ref}(\tau)^{\bot}} \ \textsc{T-alloc}$$

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

$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc} \mathsf{ref}(e_1) : \mathsf{ref}(\tau)^{\ell} \qquad \Xi \mid \Psi \vdash \tau <: \tau' \qquad \Psi \vdash \tau' \searrow \ell}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} !\, e : \tau'} \ \textsc{T-load}$$

$$\frac{\Xi \mid \Psi \mid \Gamma \vdash_{pc'} e : \tau' \qquad \Psi \vdash pc \sqsubseteq pc' \qquad \Xi \mid \Psi \vdash \tau' <: \tau}{\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : \tau} \ \textsc{T-sub}$$

3 Modal Weakest Precondition (MWP)

We refer to the Coq formalization for details not described in this document. Note that the MWP-theory is implicitly parameterized over a suitable language with expressions $e \in \mathit{Expr}$, values $v \in \mathit{Val}$, a stepping relation $(e, \sigma_1) \to (e_2, \sigma_2)$, and a state interpretation $S : \mathit{State} \to \mathit{iProp}$.

Definition 3.1 (MWP). Let $\mathcal{M} = (A, B, \mathsf{M}, \mathsf{BindCond})$ where

$$\begin{aligned} A, B &: \mathit{Type} \\ \mathsf{M} &: A \to \mathit{Masks} \to \mathbb{N} \to (B \to \mathit{iProp}) \to \mathit{iProp} \\ \mathsf{BindCond} &: A \to A \to (B \to A) \to (B \to B \to B) \to \mathsf{Prop} \end{aligned}$$

with $a \in A$ and $\mathcal{E} \in \mathit{Masks}$ then

$$\mathsf{mwp}_{\mathcal{E}}^{\mathcal{M};a}\ e\ \{\Phi\} \triangleq \forall \sigma_1, \sigma_2, v, n.\ (e, \sigma_1) \to^n (v, \sigma_2) \twoheadrightarrow S(\sigma_1) \twoheadrightarrow \mathsf{M}^a_{\mathcal{E};n}(\lambda b.\ \Phi(v, n, b) * S(\sigma_2)).$$

When omitting the mask $\mathcal{E}$ we assume it as the largest possible mask $\top$.

Definition 3.2 (MWP validity). A modality $\mathcal{M} = (A, B, \mathsf{M}, \mathsf{BindCond})$ is *valid* if

$$\forall a, \mathcal{E}, \mathcal{E}', n, \Phi, \Psi.\ \mathcal{E} \subseteq \mathcal{E}' \Rightarrow \forall b.\ \Phi(b) \twoheadrightarrow \Psi(b) \vdash \mathsf{M}^a_{\mathcal{E};n}(\Phi) \twoheadrightarrow \mathsf{M}^a_{\mathcal{E}';n}(\Psi) \qquad \text{(monotone)}$$

$$\forall a, \mathcal{E}, n, \Phi.\ \mathsf{M}^a_{\mathcal{E};0}(\Phi) \vdash \mathsf{M}^a_{\mathcal{E};n}(\Phi) \qquad \text{(introducable)}$$

$$\forall a, a', f, g, \mathcal{E}, n, m, \Phi.\ \mathsf{BindCond}(a, a', f, g) \Rightarrow$$
$$\mathsf{M}^{a'}_{\mathcal{E};n}(\lambda b.\ \mathsf{M}^{f(b)}_{\mathcal{E};m}(\lambda b'.\ \Phi(g(b, b')))) \vdash \mathsf{M}^a_{\mathcal{E};n+m}(\Phi) \qquad \text{(binding)}$$

Lemma 3.3 (M validity). Given a valid modality $\mathcal{M} = (A, B, \mathsf{M}, \mathsf{BindCond})$ then

$$\frac{\forall v, n.\ \mathsf{M}^a_{\mathcal{E};n}(\lambda b.\ \Phi(v, n, b)) \qquad e \text{ executes purely}}{\mathsf{mwp}_{\mathcal{E}}^{\mathcal{M};a}\ e\ \{\Phi\}} \ \textsc{MWP-intro}
\qquad
\frac{\mathsf{M}^a_{\mathcal{E};0}(\lambda b.\ \Phi(v, 0, b))}{\mathsf{mwp}_{\mathcal{E}}^{\mathcal{M};a}\ v\ \{\Phi\}} \ \textsc{MWP-value}$$

$$\frac{\forall v, n, b.\ \Phi(v, n, b) \twoheadrightarrow \Psi(v, n, b) \qquad \mathsf{mwp}_{\mathcal{E}}^{\mathcal{M};a}\ e\ \{\Psi\}}{\mathsf{mwp}_{\mathcal{E}}^{\mathcal{M};a}\ e\ \{\Phi\}} \ \textsc{MWP-mono}
\qquad
\frac{\mathcal{E} \subseteq \mathcal{E}' \qquad \mathsf{mwp}_{\mathcal{E}}^{\mathcal{M};a}\ e\ \{\Phi\}}{\mathsf{mwp}_{\mathcal{E}'}^{\mathcal{M};a}\ e\ \{\Phi\}} \ \textsc{MWP-mask-mono}$$

$$\frac{\mathsf{BindCond}(a, a', f, g) \qquad \mathsf{mwp}_{\mathcal{E}}^{\mathcal{M};a'}\ e\ \left\{v, n, b.\ \mathsf{mwp}_{\mathcal{E}}^{\mathcal{M};f(b)}\ K[v]\ \{w, m, b'.\ \Phi(w, n+m, g(b, b'))\}\right\}}{\mathsf{mwp}_{\mathcal{E}}^{\mathcal{M};a}\ K[e]\ \{\Phi\}} \ \textsc{MWP-bind}$$

Definition 3.4 (Atomic shift). $\mathcal{M} = (A, B, \mathsf{M}, \mathsf{BindCond})$ *supports atomic shifts at* a if

$$\forall \mathcal{E}_1, \mathcal{E}_2, n, \Phi.\ n \leq 1 \Rightarrow {}^{\mathcal{E}_1}{\Rrightarrow}^{\mathcal{E}_2}\, \mathsf{M}^a_{\mathcal{E}_2;n}(\lambda b.\ {}^{\mathcal{E}_2}{\Rrightarrow}^{\mathcal{E}_1}\, \Phi(b)) \vdash \mathsf{M}^a_{\mathcal{E}_1;n}(\Phi)$$

Definition 3.5 (Atomic Operation).

$$\mathrm{atomic}(e) \triangleq \forall \sigma, \sigma', e'.\ (\sigma, e) \to (\sigma', e') \Rightarrow e' \in \mathit{Val}$$

Definition 3.6 (Reducible Operation).

$$\mathrm{reducible}(e, \sigma) \triangleq \exists e', \sigma'.\ (\sigma, e) \to (\sigma', e')$$

Lemma 3.7 (MWP Atomic Step). Given $\mathcal{M}$ that supports atomic shifts at a then

$$\frac{\overset{\text{MWP-ATOMIC}}{{}^{\mathcal{E}}\!\!\Rrightarrow^{\mathcal{E}'} \mathsf{mwp}^{\mathcal{M};a}_{\mathcal{E}'}\ e\left\{v, n, b.\ {}^{\mathcal{E}'}\!\!\Rrightarrow^{\mathcal{E}} \Phi(v, n, b)\right\} \qquad \mathrm{atomic}(e)}}{\mathsf{mwp}^{\mathcal{M};a}_{\mathcal{E}}\ e\ \{\Phi\}}$$

Definition 3.8 (M splitting). Let $\mathbb{M}_1, \mathbb{M}_2 : \mathit{Masks} \to \mathit{iProp} \to \mathit{iProp}$ be two modalities indexed by masks. M can be split into $(\mathbb{M}_1, \mathbb{M}_2)$, written $\mathit{SplitsInto}(\mathsf{M}; \mathbb{M}_1, \mathbb{M}_2, a)$, if

$$\begin{aligned}
&\forall \mathcal{E}, n, \Phi.\ \mathbb{M}_1(\mathcal{E})\left(\mathbb{M}_2(\mathcal{E})\left(\mathsf{M}^a_{\mathcal{E};n}(\Phi)\right)\right) \vdash \mathsf{M}^a_{\mathcal{E};n+1}(\Phi)\\
&\forall \mathcal{E}, P, Q.\ P \mathrel{-\!\!*} Q \vdash \mathbb{M}_1(\mathcal{E})(P) \mathrel{-\!\!*} \mathbb{M}_1(\mathcal{E})(Q)\\
&\forall \mathcal{E}, P, Q.\ P \mathrel{-\!\!*} Q \vdash \mathbb{M}_2(\mathcal{E})(P) \mathrel{-\!\!*} \mathbb{M}_2(\mathcal{E})(Q)
\end{aligned}$$

Lemma 3.9 (Lifting). Let $a \in A$ and M a modality with $\mathit{SplitsInto}(\mathsf{M}; \mathbb{M}_1, \mathbb{M}_2, a)$ then

MWP-LIFT-STEP

$$\frac{e_1 \notin \mathit{Val} \qquad \forall \sigma_1.\ S(\sigma_1) \mathrel{-\!\!*} \mathbb{M}_1(\mathcal{E})\begin{pmatrix}\forall \sigma_2, e_2.\ (e, \sigma_1) \to (e_2, \sigma_2) \mathrel{-\!\!*} \\ \qquad \mathbb{M}_2(\mathcal{E})\left(S(\sigma_2) * \mathsf{mwp}^{\mathcal{M};a}_{\mathcal{E}}\ e_2\ \{v, n, b.\ \Phi(v, n+1, b)\}\right)\end{pmatrix}}{\mathsf{mwp}^{\mathcal{M};a}_{\mathcal{E}}\ e_1\ \{\Phi\}}$$

Definition 3.10 (MWP instance: Unary update). Let $\mathcal{M}_{\Rrightarrow} \triangleq (1, 1, \mathsf{M}, \mathsf{BindCond})$ where

$$\begin{aligned}
\mathsf{M}^a_{\mathcal{E};n}(\Phi) &\triangleq {\Rrightarrow_{\mathcal{E}}} \Phi()\\
\mathsf{BindCond}(a, a', f, g) &\triangleq \lambda_, g = id
\end{aligned}$$

Lemma 3.11 (Properties of $\mathcal{M}_{\Rrightarrow}$).

1. $\mathcal{M}_{\Rrightarrow}$ defines a valid modality.
2. $\mathcal{M}_{\Rrightarrow}$ supports atomic shifts.
3. $\mathit{SplitsInto}(\mathsf{M}; {}^{\mathcal{E}}\!\!\Rrightarrow^{\emptyset}, {}^{\emptyset}\!\!\Rrightarrow^{\mathcal{E}})$.

Lemma 3.12 (Unary update MWP always supports atomic shifts).

$${}^{\mathcal{E}_1}\!\!\Rrightarrow^{\mathcal{E}_2} \mathsf{mwp}^{\mathcal{M}_{\Rrightarrow}}_{\mathcal{E}_1}\ e\left\{v, n, b.\ {}^{\mathcal{E}_2}\!\!\Rrightarrow^{\mathcal{E}_1} \Phi(v, n, b)\right\} \mathrel{-\!\!*} \mathsf{mwp}^{\mathcal{M}_{\Rrightarrow}}_{\mathcal{E}_1}\ e\ \{\Phi\}$$

Definition 3.13 (MWP instance: Unary step-update). Let $\mathcal{M}_{\Rrightarrow\triangleright} \triangleq (1, 1, \mathsf{M}, \mathsf{BindCond})$ where

$$\begin{aligned}
\mathsf{M}^a_{\mathcal{E};n}(\Phi) &\triangleq ({}^{\mathcal{E}}\!\!\Rrightarrow^{\emptyset} \triangleright {}^{\emptyset}\!\!\Rrightarrow^{\mathcal{E}})^n {\Rrightarrow_{\mathcal{E}}} \Phi()\\
\mathsf{BindCond}(a, a', f, g) &\triangleq \lambda_, g = id
\end{aligned}$$

Lemma 3.14 (Properties of $\mathcal{M}_{\Rrightarrow\triangleright}$).

1. $\mathcal{M}_{\Rrightarrow\triangleright}$ defines a valid modality.
2. $\mathcal{M}_{\Rrightarrow\triangleright}$ supports atomic shifts.
3. $\mathit{SplitsInto}(\mathsf{M}; {}^{\mathcal{E}}\!\!\Rrightarrow^{\emptyset} \triangleright, {}^{\emptyset}\!\!\Rrightarrow^{\mathcal{E}})$.

Definition 3.15 (MWP instance: Binary update)**.** Let $\mathcal{M}_{\times\Rrightarrow} \triangleq$ (*Expr*, *Val* $\times\, \mathbb{N}$, M, BindCond) where

$$\mathsf{M}^{e}_{\mathcal{E};n}(\Phi) \triangleq \mathsf{mwp}^{\mathcal{M}_{\Rrightarrow}}_{\mathcal{E}}\ e\,\{w, m.\ \Phi(w, m)\}$$
$$\mathsf{BindCond}(e_1, e_2, f, g) \triangleq \exists K.\, e_1 = K[e_2] \wedge\ g = \lambda(v_1, n_1), (v_2, n_2).(v_2, n_1 + n_2) \wedge$$
$$\forall v, k.\, f(v, k) = K[v].$$

Lemma 3.16 (Properties of $\mathcal{M}_{\times\Rrightarrow}$)**.**

1. $\mathcal{M}_{\times\Rrightarrow}$ defines a valid modality.
2. $\forall a.\ \mathit{SplitsInto}(\mathsf{M};\ {}^{\mathcal{E}}{\Rrightarrow}^{\emptyset}, {}^{\emptyset}{\Rrightarrow}^{\mathcal{E}}, a)$.

Fact 3.17 (Unfolding MWP with $\mathcal{M}_{\times\Rrightarrow}$)**.** By unfolding the definition of MWP instantiated with $\mathcal{M}_{\Rrightarrow}$ we get:

$$\begin{aligned}
\mathsf{mwp}^{\mathcal{M}_{\times\Rrightarrow};e_2}_{\mathcal{E}}\ e_1\,\{\Phi\} &= \forall \sigma_1, \sigma_1', v, n.\, (e_1, \sigma_1) \to^n (v, \sigma_1') \twoheadrightarrow S_1(\sigma_1) \twoheadrightarrow \\
&\qquad \mathsf{M}^{\mathcal{M}_{\times\Rrightarrow};e_2}_{\mathcal{E};n}(\lambda X.\ \Phi(v, n, X) * S_1(\sigma_1')) \\
&= \forall \sigma_1, \sigma_1', v, n.\, (e_1, \sigma_1) \to^n (v, \sigma_1') \twoheadrightarrow S_1(\sigma_1) \twoheadrightarrow \\
&\qquad \mathsf{mwp}^{\mathcal{M}_{\Rrightarrow}}_{\mathcal{E}}\ e_2\,\{w, m.\ \Phi(v, n, (w, m)) * S_1(\sigma_1'))\} \\
&= \forall \sigma_1, \sigma_1', v, n.\, (e_1, \sigma_1) \to^n (v, \sigma_1') \twoheadrightarrow S_1(\sigma_1) \twoheadrightarrow \\
&\qquad \forall \sigma_2, \sigma_2', w, m.\, (e_2, \sigma_2) \to^m (w, \sigma_2') \twoheadrightarrow S_2(\sigma_2) \twoheadrightarrow \\
&\qquad\qquad \mathsf{M}^{\mathcal{M}_{\Rrightarrow}}_{\mathcal{E};m}\ (\lambda X.\ \Phi(v, n, (w, m)) * S_1(\sigma_1') * S_2(\sigma_2')) \\
&= \forall \sigma_1, \sigma_1', v, n.\, (e_1, \sigma_1) \to^n (v, \sigma_1') \twoheadrightarrow S_1(\sigma_1) \twoheadrightarrow \\
&\qquad \forall \sigma_2, \sigma_2', w, m.\, (e_2, \sigma_2) \to^m (w, \sigma_2') \twoheadrightarrow S_2(\sigma_2) \twoheadrightarrow \\
&\qquad\qquad \Rrightarrow_{\mathcal{E}}\ (\Phi(v, n, (w, m)) * S_1(\sigma_1') * S_2(\sigma_2'))
\end{aligned}$$

Lemma 3.18 (Unary update MWP implies binary update MWP)**.**

$$\mathsf{mwp}^{\mathcal{M}_{\Rrightarrow}}_{\mathcal{E}}\ e_1\left\{v, n.\ \mathsf{mwp}^{\mathcal{M}_{\Rrightarrow}}_{\mathcal{E}}\ e_2\,\{w, m.\ \Phi(v, n, (w, m))\}\right\} \twoheadrightarrow \mathsf{mwp}^{\mathcal{M}_{\times\Rrightarrow};e_2}_{\mathcal{E}}\ e_1\,\{\Phi\}$$
$$\mathsf{mwp}^{\mathcal{M}_{\Rrightarrow}}_{\mathcal{E}}\ e_2\left\{w, m.\ \mathsf{mwp}^{\mathcal{M}_{\Rrightarrow}}_{\mathcal{E}}\ e_1\,\{v, n.\ \Phi(v, n, (w, m))\}\right\} \twoheadrightarrow \mathsf{mwp}^{\mathcal{M}_{\times\Rrightarrow};e_2}_{\mathcal{E}}\ e_1\,\{\Phi\}$$

Lemma 3.19 (Binary update MWP always supports shifts)**.**

$${}^{\mathcal{E}_1}{\Rrightarrow}^{\mathcal{E}_2}\ \mathsf{mwp}^{\mathcal{M}_{\Rrightarrow};e_2}_{\mathcal{E}_1}\ e_1\left\{v, n, b.\ {}^{\mathcal{E}_2}{\Rrightarrow}^{\mathcal{E}_1}\,\Phi(v, n, b)\right\} \twoheadrightarrow \mathsf{mwp}^{\mathcal{M}_{\times\Rrightarrow};e_2}_{\mathcal{E}_1}\ e_1\,\{\Phi\}$$

Definition 3.20 (MWP instance: Binary step-update)**.** Let $\mathcal{M}_I \triangleq (\mathbb{N}, 1, \mathsf{M}, \mathsf{BindCond})$ where

$$\mathsf{M}^{m}_{\mathcal{E};n}(\Phi) \triangleq ({}^{\mathcal{E}}{\Rrightarrow}^{\emptyset} \triangleright {}^{\emptyset}{\Rrightarrow}^{\mathcal{E}})^{n+m} \Rrightarrow_{\mathcal{E}} \Phi()$$
$$\mathsf{BindCond}(n, m, f, g) \triangleq m \leq n \wedge \forall x, f(x) = n - m \wedge \lambda\_, g = id.$$

Let $\mathcal{M}_{\times\Rrightarrow\triangleright} \triangleq$ (*Expr*, *Val* $\times\, \mathbb{N}$, M, BindCond) where

$$\mathsf{M}^{e}_{\mathcal{E};n}(\Phi) \triangleq \mathsf{mwp}^{\mathcal{M}_I;n}_{\mathcal{E}}\ e\,\{w, m.\ \Phi(w, m)\}$$
$$\mathsf{BindCond}(e_1, e_2, f, g) \triangleq \exists K.\, e_1 = K[e_2] \wedge\ g = \lambda(v_1, n_1), (v_2, n_2).(v_2, n_1 + n_2) \wedge$$
$$\forall v, k.\, f(v, k) = K[v].$$

Lemma 3.21 (Properties of $\mathcal{M}_{\times\Rrightarrow\triangleright}$)**.**

1. $\mathcal{M}_{\times\Rrightarrow\triangleright}$ is a valid MWP-modality.
2. $\forall a.\ \mathit{SplitsInto}(\mathsf{M};\ {}^{\mathcal{E}}{\Rrightarrow}^{\emptyset} \triangleright, {}^{\emptyset}{\Rrightarrow}^{\mathcal{E}}, a)$.

Fact 3.22 (Unfolding MWP with $\mathcal{M}_{\times\Rrightarrow\triangleright}$)**.** By unfolding the definition of MWP instantiated $\mathcal{M}_{\times\Rrightarrow\triangleright}$ we get:

$$\begin{aligned}
\mathsf{mwp}_{\mathcal{E}}^{\mathcal{M}_{\times\Rrightarrow\triangleright};e_2}\ e_1\ \{\Phi\} &= \forall \sigma_1, \sigma_1', v, n.\ (e_1, \sigma_1) \to^n (v, \sigma_1') \twoheadrightarrow S_1(\sigma_1) \twoheadrightarrow \\
&\qquad \mathsf{M}_{\mathcal{E};n}^{\mathcal{M}_{\times\Rrightarrow\triangleright};e_2}(\lambda X.\ \Phi(v, n, X) * S_1(\sigma_1')) \\
&= \forall \sigma_1, \sigma_1', v, n.\ (e_1, \sigma_1) \to^n (v, \sigma_1') \twoheadrightarrow S_1(\sigma_1) \twoheadrightarrow \\
&\qquad \mathsf{mwp}_{\mathcal{E}}^{\mathcal{M}_I;n}\ e_2\ \{w, m.\ \Phi(v, n, (w, m)) * S_1(\sigma_1')\} \\
&= \forall \sigma_1, \sigma_1', v, n.\ (e_1, \sigma_1) \to^n (v, \sigma_1') \twoheadrightarrow S_1(\sigma_1) \twoheadrightarrow \\
&\qquad \forall \sigma_2, \sigma_2', w, m.\ (e_2, \sigma_2) \to^m (w, \sigma_2') \twoheadrightarrow S_2(\sigma_2) \twoheadrightarrow \\
&\qquad\qquad \mathsf{M}_{\mathcal{E};m}^{\mathcal{M}_I;n}\ ((\lambda X.\ \Phi(v, n, (w, m)) * S_1(\sigma_1') * S_2(\sigma_2'))) \\
&= \forall \sigma_1, \sigma_1', v, n.\ (e_1, \sigma_1) \to^n (v, \sigma_1') \twoheadrightarrow S_1(\sigma_1) \twoheadrightarrow \\
&\qquad \forall \sigma_2, \sigma_2', w, m.\ (e_2, \sigma_2) \to^m (w, \sigma_2') \twoheadrightarrow S_2(\sigma_2) \twoheadrightarrow \\
&\qquad\qquad ({}^{\mathcal{E}}{\Rrightarrow}^{\emptyset} \triangleright {}^{\emptyset}{\Rrightarrow}^{\mathcal{E}})^{n+m} \Rrightarrow_{\mathcal{E}} (\Phi(v, n, (w, m)) * S_1(\sigma_1') * S_2(\sigma_2'))
\end{aligned}$$

Lemma 3.23 (Unary step-update MWP implies binary step-update MWP)**.**

$$\mathsf{mwp}_{\mathcal{E}}^{\mathcal{M}_{\Rrightarrow\triangleright}}\ e_1 \left\{ v, n.\ \mathsf{mwp}_{\mathcal{E}}^{\mathcal{M}_{\Rrightarrow\triangleright}}\ e_2\ \{w, m.\ \Phi(v, n, (w, m))\} \right\} \twoheadrightarrow \mathsf{mwp}_{\mathcal{E}}^{\mathcal{M}_{\times\Rrightarrow\triangleright};e_2}\ e_1\ \{\Phi\}$$
$$\mathsf{mwp}_{\mathcal{E}}^{\mathcal{M}_{\Rrightarrow\triangleright}}\ e_2 \left\{ w, m.\ \mathsf{mwp}_{\mathcal{E}}^{\mathcal{M}_{\Rrightarrow\triangleright}}\ e_1\ \{v, n.\ \Phi(v, n, (w, m))\} \right\} \twoheadrightarrow \mathsf{mwp}_{\mathcal{E}}^{\mathcal{M}_{\times\Rrightarrow\triangleright};e_2}\ e_1\ \{\Phi\}$$

Lemma 3.24 (Double atomicity of binary step-update MWP)**.** If $\mathrm{atomic}(e_1)$ and $\mathrm{atomic}(e_2)$ then

$$^{\mathcal{E}_1}{\Rrightarrow}^{\mathcal{E}_2}\ \mathsf{mwp}_{\mathcal{E}_2}^{\mathcal{M}_{\Rrightarrow\triangleright}}\ e_1 \left\{ v, n.\ \mathsf{mwp}_{\mathcal{E}_2}^{\mathcal{M}_{\Rrightarrow\triangleright}}\ e_2 \left\{ w, m.\ {}^{\mathcal{E}_2}{\Rrightarrow}^{\mathcal{E}_1}\ \Phi(v, n, (w, m)) \right\} \right\} \twoheadrightarrow \mathsf{mwp}_{\mathcal{E}_1}^{\mathcal{M}_{\times\Rrightarrow\triangleright};e_2}\ e_1\ \{\Phi\}$$
$$^{\mathcal{E}_1}{\Rrightarrow}^{\mathcal{E}_2}\ \mathsf{mwp}_{\mathcal{E}_2}^{\mathcal{M}_{\Rrightarrow\triangleright}}\ e_2 \left\{ w, m.\ \mathsf{mwp}_{\mathcal{E}_2}^{\mathcal{M}_{\Rrightarrow\triangleright}}\ e_1 \left\{ v, n.\ {}^{\mathcal{E}_2}{\Rrightarrow}^{\mathcal{E}_1}\ \Phi(v, n, (w, m)) \right\} \right\} \twoheadrightarrow \mathsf{mwp}_{\mathcal{E}_1}^{\mathcal{M}_{\times\Rrightarrow\triangleright};e_2}\ e_1\ \{\Phi\}$$

Lemma 3.25 (Binary update MWP implies binary step-update MWP)**.** Let

$$\mathrm{reduces}(e, S, \mathcal{E}) \triangleq \forall \sigma.\ S(\sigma)\ {}^{\mathcal{E}}{\Rrightarrow\!\!\ast}^{\emptyset}\ \mathrm{reducible}(e, \sigma).$$

Then

$$\begin{aligned}
&(\mathrm{reduces}(e_1, S_1, \mathcal{E}_1) \vee \mathrm{reduces}(e_2, S_2, \mathcal{E}_1)) \wedge \\
&\quad \left({}^{\mathcal{E}_1}{\Rrightarrow}^{\mathcal{E}_2} \triangleright \mathsf{mwp}_{\mathcal{E}_2}^{\mathcal{M}_{\times\Rrightarrow};e_2}\ e_1 \left\{ v, n, b.\ {}^{\mathcal{E}_2}{\Rrightarrow}^{\mathcal{E}_1}\ \Phi(v, n, b) \right\} \right) \twoheadrightarrow \mathsf{mwp}_{\mathcal{E}_1}^{\mathcal{M}_{\times\Rrightarrow\triangleright};e_2}\ e_1\ \{\Phi\}.
\end{aligned}$$

Theorem 3.26 (Adequacy of binary step-update MWP)**.** Let φ be a first-order predicate over values. Suppose

$$\mathsf{mwp}_{\mathcal{E}}^{\mathcal{M}_{\times\Rrightarrow\triangleright};e_2}\ e_1\ \{\varphi\}$$

is derivable. Given $S_1(\sigma_1)$ and $S_2(\sigma_2)$, if we have $(\sigma_1, e_1) \to^{n_1} (\sigma_1, v_1)$ and $(\sigma_2, e_2) \to^{n_2} (\sigma_2', v_2)$ then $\varphi(v_1, n_1, v_2, n_2)$ holds at the meta-level.

3.1 Language-level lemmas

By instantiating the MWP-theory with λ_{sec} and state interpretation $\lambda\sigma.\ \boxed{\bullet\, \sigma}^{\gamma}$ with $\iota \hookrightarrow v \triangleq \boxed{\circ\, [\iota \mapsto v]}^{\gamma}$ for modelling the heap we get the following lemmas for interaction with the heap.

Lemma 3.27 (Properties of unary update MWP with λ_{sec}).

1. $\forall \iota.\ \iota \hookrightarrow v \twoheadrightarrow Q\ \iota \vdash \mathsf{mwp}_{\mathcal{E}}^{\mathcal{M}_{\Rrightarrow}}\ \mathsf{ref}(v)\ \{v.\ Q\}$
2. $\iota \hookrightarrow v * (\iota \hookrightarrow v \twoheadrightarrow Q\ v) \vdash \mathsf{mwp}_{\mathcal{E}}^{\mathcal{M}_{\Rrightarrow}}\ !\,\iota\ \{v.\ Q\}$
3. $\iota \hookrightarrow v * (\iota \hookrightarrow w \twoheadrightarrow Q\ ()) \vdash \mathsf{mwp}_{\mathcal{E}}^{\mathcal{M}_{\Rrightarrow}}\ \iota \leftarrow w\ \{v.\ Q\}$

Lemma 3.28 (Properties of unary step-taking update MWP with λ_{sec})**.**

1. $\triangleright \forall \iota.\, \iota \hookrightarrow v \twoheadrightarrow Q\ \iota \vdash \mathsf{mwp}_{\mathcal{E}}^{\mathcal{M}_{\Rrightarrow\triangleright}}\ \mathsf{ref}(v)\ \{v.\ Q\}$
2. $\triangleright \iota \hookrightarrow v * \triangleright(\iota \hookrightarrow v \twoheadrightarrow Q\ v) \vdash \mathsf{mwp}_{\mathcal{E}}^{\mathcal{M}_{\Rrightarrow\triangleright}}\ !\,\iota\ \{v.\ Q\}$
3. $\triangleright \iota \hookrightarrow v * \triangleright(\iota \hookrightarrow w \twoheadrightarrow Q\ ()) \vdash \mathsf{mwp}_{\mathcal{E}}^{\mathcal{M}_{\Rrightarrow\triangleright}}\ \iota \leftarrow w\ \{v.\ Q\}$

4 Logical Relations

The binary value relation is an Iris relation of type $\mathit{Rel} \triangleq \mathit{Val} \times \mathit{Val} \to \mathit{iProp}_{\square}$. Similarly, the unary value relation is an Iris predicate of type $\mathit{Pred} \triangleq \mathit{Val} \to \mathit{iProp}_{\square}$.

Both the unary and binary logical relation is implicitly quantified over a lattice $\mathcal{L}$ and an observer/attacker label ζ. The environment $\rho : \mathit{Lvar} \to \mathcal{L}$ maps label variables to semantic labels from $\mathcal{L}$ and Θ is a semantic type environment for type variables, as is usual for interpretations of languages with parametric polymorphism. However, for every type variable we keep both a binary relation and two unary relations, one for each of the two sides:

$$\Theta : \mathit{Tvar} \to \mathit{Rel} \times \mathit{Pred} \times \mathit{Pred}.$$

We use $\Theta_{\mathsf{L}}, \Theta_{\mathsf{R}} : \mathit{Tvar} \to \mathit{Pred}$ as shorthand for $\pi_2 \circ \Theta$ and $\pi_3 \circ \Theta$, respectively, where $\pi_i(x)$ denotes the ith projection of x. We will use

$$\mathsf{mwp}_{\mathcal{E}}\ e_1 \sim e_2\ \{v, w.\ Q\}$$

as shorthand for $\mathsf{mwp}_{\mathcal{E}}^{\mathcal{M}_{\times \Rrightarrow\triangleright};e_2}\ e_1\ \{v, \_, (w, \_).\ Q\}$.

Definition 4.1 (Label interpretation)**.**

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

Definition 4.2 (Unary value interpretation).

$$
\begin{aligned}
[\![\alpha]\!]^{\rho}_{\Delta} &\triangleq \Delta(\alpha) \\
[\![1]\!]^{\rho}_{\Delta}(v) &\triangleq v = () \\
[\![\mathbb{B}]\!]^{\rho}_{\Delta}(v) &\triangleq v \in \{\mathsf{true}, \mathsf{false}\} \\
[\![\mathbb{N}]\!]^{\rho}_{\Delta}(v) &\triangleq v \in \mathbb{N} \\
[\![\tau_1 \times \tau_2]\!]^{\rho}_{\Delta}(v) &\triangleq \exists v_1, v_2.\, v = (v_1, v_2) * [\![\tau_1]\!]^{\rho}_{\Delta}(v_1) * [\![\tau_2]\!]^{\rho}_{\Delta}(v_2) \\
[\![\tau_1 + \tau_2]\!]^{\rho}_{\Delta}(v) &\triangleq \bigvee_{i \in \{1,2\}} \exists w.\, v = \mathsf{inj}_i\, w * [\![\tau_i]\!]^{\rho}_{\Delta}(w) \\
[\![\tau_1 \xrightarrow{\ell_e} \tau_2]\!]^{\rho}_{\Delta}(v) &\triangleq \square\, (\forall w.\, [\![\tau_1]\!]^{\rho}_{\Delta}(w) \mathrel{-\!\!*} \mathcal{E}_{\ell_e}[\![\tau_2]\!]^{\rho}_{\Delta}(v\ w)) \\
[\![\forall_{\ell_e}\, \alpha.\, \tau]\!]^{\rho}_{\Delta}(v) &\triangleq \square \left(\forall \Phi : \mathit{Pred}.\, \mathcal{E}_{\ell_e}[\![\tau]\!]^{\rho}_{\Delta, \alpha \mapsto \Phi}(v\ \_)\right) \\
[\![\exists \alpha.\, \tau]\!]^{\rho}_{\Delta}(v) &\triangleq \square \left(\exists \Phi : \mathit{Pred}.\, \exists w.\, v = \mathsf{pack}\, w * [\![\tau]\!]^{\rho}_{\Delta, \alpha \mapsto \Phi}(w)\right) \\
[\![\forall_{\ell_e}\, \kappa.\, \tau]\!]^{\rho}_{\Delta}(v) &\triangleq \square \left(\forall l \in \mathcal{L}.\, \mathcal{E}_{\ell_e}[\![\tau]\!]^{\rho, \kappa \mapsto l}_{\Delta}(v\ \_)\right) \\
[\![\mu \alpha.\, \tau]\!]^{\rho}_{\Delta} &\triangleq \mu \Phi : \mathit{Pred}.\, \lambda v. \exists w.\, v = \mathsf{fold}\, w * \triangleright [\![\tau]\!]^{\rho}_{\Delta, \alpha \mapsto f}(w) \\
[\![\mathsf{ref}(t^{\ell})]\!]^{\rho}_{\Delta}(v) &\triangleq \exists \iota, \mathcal{N}.\, v = \iota * \mathcal{R}(\Delta, \rho, \iota, \ell, \mathcal{N}) \\
\mathcal{R}(\Delta, \rho, \iota, \ell, \mathcal{N}) &\triangleq \begin{cases}
\square \forall \mathcal{E}.\, \mathcal{N}^{\uparrow} \subseteq \mathcal{E} \Rightarrow & \\
\quad \left({}^{\mathcal{E}}\!\Rrightarrow^{\mathcal{E} \setminus \mathcal{N}^{\uparrow}} \triangleright \begin{pmatrix} \exists w.\, \iota \mapsto_i w * [\![\tau]\!]^{\rho}_{\Delta}(w) * \\ ((\triangleright \iota \mapsto_i w * [\![\tau]\!]^{\rho}_{\Delta}(w)) \;{}^{\mathcal{E} \setminus \mathcal{N}^{\uparrow}}\!\!\Rrightarrow\!\!\ast^{\mathcal{E}}\ \mathsf{True}) \end{pmatrix} \right) & \text{if } [\![\ell]\!]_{\rho} \sqsubseteq \zeta \\
& \\
\square \forall \mathcal{E}.\, \mathcal{N}^{\uparrow} \subseteq \mathcal{E} \Rightarrow & \\
\quad \left({}^{\mathcal{E}}\!\Rrightarrow^{\mathcal{E} \setminus \mathcal{N}^{\uparrow}} \triangleright \begin{pmatrix} \exists w.\, \iota \mapsto_i w * [\![\tau]\!]^{\rho}_{\Delta}(w) * \\ ((\triangleright \exists w'.\, \iota \mapsto_i w' * [\![\tau]\!]^{\rho}_{\Delta}(w')) \;{}^{\mathcal{E} \setminus \mathcal{N}^{\uparrow}}\!\!\Rrightarrow\!\!\ast^{\mathcal{E}}\ \mathsf{True}) \end{pmatrix} \right) & \text{if } [\![\ell]\!]_{\rho} \not\sqsubseteq \zeta
\end{cases} \\
[\![t^{\ell}]\!]^{\rho}_{\Delta}(v) &\triangleq [\![t]\!]^{\rho}_{\Delta}(v)
\end{aligned}
$$

Definition 4.3 (Unary expression interpretation).

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

Definition 4.4 (Unary environment interpretation).

$$
\begin{aligned}
\mathcal{G}[\![\cdot]\!]^{\rho}_{\Delta}(\epsilon) &\triangleq \mathsf{True} \\
\mathcal{G}[\![\Gamma, x : \tau]\!]^{\rho}_{\Delta}(\vec{v} w) &\triangleq \mathcal{G}[\![\Gamma]\!]^{\rho}_{\Delta}(\vec{v}) * [\![\tau]\!]^{\rho}_{\Delta}(w)
\end{aligned}
$$

Definition 4.5 (Unary semantic typing).

$$\Xi \mid \Psi \mid \Gamma \vDash_{pc} e : \tau \triangleq \square \begin{pmatrix} \forall \Delta, \rho, \vec{v}.\, \mathrm{dom}(\Xi) \subseteq \mathrm{dom}(\Delta) * \mathrm{dom}(\Psi) \subseteq \mathrm{dom}(\rho) \mathrel{-\!\!*} \\ \mathcal{G}[\![\Gamma]\!]^{\rho}_{\Delta}(\vec{v}) \mathrel{-\!\!*} \mathcal{E}_{pc}[\![\tau]\!]^{\rho}_{\Delta}(e[\vec{v}/\vec{x}]) \end{pmatrix}$$

Lemma 4.6 (Unary semantic subtyping). If $\mathrm{dom}(\Xi) \subseteq \mathrm{dom}(\Delta)$ and $\mathrm{dom}(\Psi) \subseteq \mathrm{dom}(\rho)$ then

$$\Xi \mid \Psi \vdash \tau_1 <: \tau_2 \Rightarrow [\![\tau_1]\!]^{\rho}_{\Delta}(v) \mathrel{-\!\!*} [\![\tau_2]\!]^{\rho}_{\Delta}(v)$$

Theorem 4.7 (Unary fundamental theorem).

$$\Xi \mid \Psi \mid \Gamma \vdash_{pc} e : \tau \Rightarrow \Xi \mid \Psi \mid \Gamma \vDash_{pc} e : \tau$$

Definition 4.8 (Binary value interpretation).

$$
\begin{aligned}
\llbracket \alpha \rrbracket^{\rho}_{\Theta} &\triangleq \pi_1\left(\Theta(\alpha)\right) \\
\llbracket 1 \rrbracket^{\rho}_{\Theta}(v, v') &\triangleq v = v' = () \\
\llbracket \mathbb{B} \rrbracket^{\rho}_{\Theta}(v, v') &\triangleq v = v' \in \{\mathsf{true}, \mathsf{false}\} \\
\llbracket \mathbb{N} \rrbracket^{\rho}_{\Theta}(v, v') &\triangleq v = v' \in \mathbb{N} \\
\llbracket \tau_1 \times \tau_2 \rrbracket^{\rho}_{\Theta}(v, v') &\triangleq \exists v_1, v_2, v_1', v_2'.\, v = (v_1, v_2) * v' = (v_1', v_2') * \llbracket \tau_1 \rrbracket^{\rho}_{\Theta}(v_1, v_1') * \llbracket \tau_2 \rrbracket^{\rho}_{\Theta}(v_2, v_2') \\
\llbracket \tau_1 + \tau_2 \rrbracket^{\rho}_{\Theta}(v, v') &\triangleq \bigvee_{i \in \{1,2\}} \exists w, w'.\, v = \mathsf{inj}_i\, w * v' = \mathsf{inj}_i\, w' * \llbracket \tau_i \rrbracket^{\rho}_{\Theta}(w, w') \\
\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') \twoheadrightarrow \mathcal{E}\llbracket \tau_2 \rrbracket^{\rho}_{\Theta}(v\ w, v'\ w')\right) \\
&\qquad * \llbracket \tau_1 \xrightarrow{\ell_e} \tau_2 \rrbracket^{\rho}_{\Theta_{\mathsf{L}}}(v) * \llbracket \tau_1 \xrightarrow{\ell_e} \tau_2 \rrbracket^{\rho}_{\Theta_{\mathsf{R}}}(v') \\
\llbracket \forall_{\ell_e}\, \alpha.\, \tau \rrbracket^{\rho}_{\Theta}(v, v') &\triangleq \Box \Big(\forall \Phi : \mathit{Rel}.\, \forall \Phi_{\mathsf{L}}, \Phi_{\mathsf{R}} : \mathit{Pred}. \\
&\qquad\qquad \Box \left(\forall v, v'.\, \Phi(v, v') \twoheadrightarrow \Phi_{\mathsf{L}}(v) * \Phi_{\mathsf{R}}(v')\right) \twoheadrightarrow \mathcal{E}\llbracket \tau \rrbracket^{\rho}_{\Theta, \alpha \mapsto (\Psi, \Phi_{\mathsf{L}}, \Phi_{\mathsf{R}})}(v\ \_, v'\ \_)\Big) \\
&\qquad * \llbracket \forall_{\ell_e}\, \alpha.\, \tau \rrbracket^{\rho}_{\Theta_{\mathsf{L}}}(v) * \llbracket \forall_{\ell_e}\, \alpha.\, \tau \rrbracket^{\rho}_{\Theta_{\mathsf{R}}}(v') \\
\llbracket \exists \alpha.\, \tau \rrbracket^{\rho}_{\Delta}(v, v') &\triangleq \Box \Big(\exists \Phi : \mathit{Rel}.\, \exists \Phi_{\mathsf{L}}, \Phi_{\mathsf{R}} : \mathit{Pred}.\, \Box \left(\forall v, v'.\, \Phi(v, v') \twoheadrightarrow \Phi_{\mathsf{L}}(v) * \Phi_{\mathsf{R}}(v')\right) * \\
&\qquad\qquad \exists w, w'.\, v = \mathsf{pack}\, w * v' = \mathsf{pack}\, w' * \llbracket \tau \rrbracket^{\rho}_{\Delta, \alpha \mapsto (\Phi, \Phi_{\mathsf{L}}, \Phi_{\mathsf{R}})}(w, w')\Big) \\
\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_{\mathsf{L}}}(v) * \llbracket \forall_{\ell_e}\, \kappa.\, \tau \rrbracket^{\rho}_{\Theta_{\mathsf{R}}}(v') \\
\llbracket \mu \alpha.\, \tau \rrbracket^{\rho}_{\Theta} &\triangleq \mu \Phi : \mathit{Rel}.\, \lambda(v, v').\exists w, w'.\, v = \mathsf{fold}\, w * v' = \mathsf{fold}\, w' \\
&\qquad * \triangleright \llbracket \tau \rrbracket^{\rho}_{\Theta, \alpha \mapsto (f, \llbracket \mu\alpha.\,\tau \rrbracket^{\rho}_{\Theta_{\mathsf{L}}}, \llbracket \mu\alpha.\,\tau \rrbracket^{\rho}_{\Theta_{\mathsf{R}}})}(w, w') \\
\llbracket \mathsf{ref}(\tau) \rrbracket^{\rho}_{\Theta}(v, v') &\triangleq \exists \iota, \iota'.\, v = \iota * v' = \iota' * \boxed{\exists w, w'.\, \iota \mapsto_{\mathsf{L}} w * \iota' \mapsto_{\mathsf{R}} w' * \llbracket \tau \rrbracket^{\rho}_{\Theta}(w, w')}^{\mathcal{N}_{root}.(\iota, \iota')} \\
\llbracket t^{\ell} \rrbracket^{\rho}_{\Theta}(v, v') &\triangleq \begin{cases} \llbracket t \rrbracket^{\rho}_{\Theta}(v, v') & \text{if } \llbracket \ell \rrbracket_{\rho} \sqsubseteq \zeta \\ \llbracket t \rrbracket^{\rho}_{\Theta_{\mathsf{L}}}(v) * \llbracket t \rrbracket^{\rho}_{\Theta_{\mathsf{R}}}(v') & \text{if } \llbracket \ell \rrbracket_{\rho} \not\sqsubseteq \zeta \end{cases}
\end{aligned}
$$

Definition 4.9 (Binary expression interpretation).

$$\mathcal{E}\llbracket \tau \rrbracket^{\rho}_{\Theta}(e, e') \triangleq \mathsf{mwp}\, e_1 \sim e_2\, \{\llbracket \tau \rrbracket^{\rho}_{\Theta}\}$$

Definition 4.10 (Binary environment interpretation).

$$
\begin{aligned}
\mathcal{G}\llbracket \cdot \rrbracket^{\rho}_{\Theta}(\epsilon, \epsilon) &\triangleq \mathsf{True} \\
\mathcal{G}\llbracket \Gamma, x : \tau \rrbracket^{\rho}_{\Theta}(\vec{v_1} w_1, \vec{v_2} w_2) &\triangleq \mathcal{G}\llbracket \Gamma \rrbracket^{\rho}_{\Theta}(\vec{v_1}, \vec{v_2}) * \llbracket \tau \rrbracket^{\rho}_{\Theta}(w_1, w_2)
\end{aligned}
$$

Definition 4.11 (Binary environment coherence).

$$\mathit{Coh}(\Theta) \triangleq \mathop{\Huge\ast}_{(\Phi, \Phi_{\mathsf{L}}, \Phi_{\mathsf{R}}) \in \Theta} \Box \left(\forall v_{\mathsf{L}}, v_{\mathsf{R}}.\, \Phi(v_{\mathsf{L}}, v_{\mathsf{R}}) \twoheadrightarrow \Phi_{\mathsf{L}}(v_{\mathsf{L}}) * \Phi_{\mathsf{R}}(v_{\mathsf{R}})\right)$$

Definition 4.12 (Binary semantic typing).

$$\Xi \mid \Psi \mid \Gamma \vDash e_{\mathsf{L}} \approx_{\zeta} e_{\mathsf{R}} : \tau \triangleq \Box \begin{pmatrix} \forall \Theta, \rho, \vec{v_{\mathsf{L}}}, \vec{v_{\mathsf{R}}}.\, \mathrm{dom}(\Xi) \subseteq \mathrm{dom}(\Theta) * \mathrm{dom}(\Psi) \subseteq \mathrm{dom}(\rho) \twoheadrightarrow \\ \mathit{Coh}(\Theta) * \mathcal{G}\llbracket \Gamma \rrbracket^{\rho}_{\Theta}(\vec{v_{\mathsf{L}}}, \vec{v_{\mathsf{R}}}) \twoheadrightarrow \mathcal{E}\llbracket \tau \rrbracket^{\rho}_{\Theta}(e_{\mathsf{L}}[\vec{v_{\mathsf{L}}}/\vec{x}], e_{\mathsf{R}}[\vec{v_{\mathsf{R}}}/\vec{x}]) \end{pmatrix}$$

Lemma 4.13 (Binary semantic subtyping). If $\mathrm{dom}(\Xi) \subseteq \mathrm{dom}(\Theta)$ and $\mathrm{dom}(\Psi) \subseteq \mathrm{dom}(\rho)$ then

$$\Xi \mid \Psi \vdash \tau_1 <: \tau_2 \Rightarrow \llbracket \tau_1 \rrbracket^{\rho}_{\Delta}(v_{\mathsf{L}}, v_{\mathsf{R}}) \twoheadrightarrow \llbracket \tau_2 \rrbracket^{\rho}_{\Delta}(v_{\mathsf{L}}, v_{\mathsf{R}})$$

Lemma 4.14 (Binary-unary subsumption).

$$\mathit{Coh}(\Theta) * \llbracket \tau \rrbracket^{\rho}_{\Theta}(v_{\mathsf{L}}, v_{\mathsf{R}}) \twoheadrightarrow \llbracket \tau \rrbracket^{\rho}_{\Theta_{\mathsf{L}}}(v_{\mathsf{L}}) * \llbracket \tau \rrbracket^{\rho}_{\Theta_{\mathsf{R}}}(v_{\mathsf{R}})$$

Theorem 4.15 (Binary fundamental theorem)**.**

$$\Xi \,|\, \Psi \,|\, \Gamma \vdash_{pc} e : \tau \Rightarrow \Xi \,|\, \Psi \,|\, \Gamma \vDash e \approx_\zeta e : \tau$$

Theorem 4.16 (Termination-Insensitive Noninterference)**.** Let $\top$ and $\bot$ be labels drawn from a join-semilattice such that $\bot \sqsubseteq \zeta$ and $\top \not\sqsubseteq \zeta$. If

$$\begin{aligned}&\cdot \,|\, \cdot \,|\, x : \mathbb{B}^\top \vdash_\bot e : \mathbb{B}^\bot,\\ &\cdot \,|\, \cdot \,|\, \cdot \vdash_\bot v_1 : \mathbb{B}^\top, \text{ and } \cdot \,|\, \cdot \,|\, \cdot \vdash_\bot v_2 : \mathbb{B}^\top\end{aligned}$$

then

$$(\emptyset, e[v_1/x]) \rightarrow^* (\sigma_1, v_1') \wedge (\emptyset, e[v_2/x]) \rightarrow^* (\sigma_2, v_2') \Rightarrow v_1' = v_2'.$$