A Relational Modal Logic for Higher-Order Stateful ADTs
(Technical Appendix)

Derek Dreyer
dreyer@mpi-sws.org
Georg Neis
neis@mpi-sws.org
Andreas Rossberg
rossberg@mpi-sws.org
Lars Birkedal
birkedal@itu.dk

July 16, 2009

Contents
1 The Language \( \mu \) ................................. 2
1.1 Syntax .................................... 2
1.2 Dynamic Semantics ........................ 2
1.3 Static Semantics ............................ 4
1.4 Contexts and Contextual Equivalence ... 5
2 LADR ........................................ 7
2.1 Well-formedness ............................ 7
2.2 Additional Inference Rules ................ 9
2.3 Model .................................... 10
2.4 Properties ................................. 11
3 Soundness of the Logical Relation .......... 12
4 Examples .................................. 16
4.1 Name Generators ........................... 16
4.2 Landin’s Knot ............................. 18

This appendix is a supplement to the paper. Things defined in the paper are not repeated here.
1 The Language \( \text{Ful} \)

1.1 Syntax

**Types**  
\[ \tau ::= \alpha \mid \text{unit} \mid \text{int} \mid \text{bool} \mid \tau_1 \times \tau_2 \mid \tau_1 + \tau_2 \mid \tau_1 \rightarrow \tau_2 \mid \forall \alpha. \tau \mid \exists \alpha. \tau \mid \mu \alpha. \tau \mid \text{ref} \tau \]

**Prim Ops**  
\[ o ::= + \mid - \mid = \mid < \mid \leq \mid \ldots \]

**Expressions**  
\[ e ::= x \mid \{I \} \mid t \mid n \mid o(e_1, \ldots, e_n) \mid \text{true} \mid \text{false} \mid \text{if} \ e \text{ then } e_1 \text{ else } e_2 \mid (e_1, e_2) \mid \text{fst } e \mid \text{snd } e \mid \text{inl } e \mid \text{inr } e \mid \text{case } e \text{ of } \text{inl } x_1 \Rightarrow e_1 \mid \text{inr } x_2 \Rightarrow e_2 \mid \lambda x : \tau. e \mid e_1 \text{ and } e_2 \mid \Lambda \alpha. e \mid e \tau \mid \text{pack } \tau_1, e \text{ as } \exists \alpha. \tau \mid \text{unpack } e \text{ as } \alpha, x \text{ in } e_2 \mid \text{roll } e \mid \text{unroll } e \mid \text{ref } E \mid e \tau \mid \text{ref } E \mid e \tau \mid \text{inl } e \mid \text{inr } e \mid \text{case } E \text{ of } \text{inl } x_1 \Rightarrow e_1 \mid \text{inr } x_2 \Rightarrow e_2 \mid E \mid v \mid E \mid E \tau \mid \text{pack } \tau_1, E \text{ as } \exists \alpha. \tau \mid \text{unpack } E \text{ as } \alpha, x \text{ in } e_2 \mid \text{roll } E \mid \text{unroll } E \mid \text{ref } E \mid !E \mid E ::= e \mid v ::= E \mid E == e \mid v == E \]

**Values**  
\[ v ::= \{I \} \mid o(v_1, \ldots, v_{i-1}, E, v_{i+1}, \ldots, e_n) \mid \text{true} \mid \text{false} \mid \langle v_1, v_2 \rangle \mid \text{inl } v \mid \text{inr } v \mid \lambda x : \tau. e \mid \Lambda \alpha. e \mid \text{pack } \tau_1, v \text{ as } \exists \alpha. \tau \mid \text{roll } v \]

**Evaluation Contexts**  
\[ E ::= [\_] \mid o(E_1, \ldots, E_{i-1}, E, E_{i+1}, \ldots, E_n) \mid \text{true} \mid \text{false} \mid \langle E_1, E_2 \rangle \mid \text{fst } E \mid \text{snd } E \mid \text{inl } E \mid \text{inr } E \mid \text{case } E \text{ of } \text{inl } x_1 \Rightarrow E_1 \mid \text{inr } x_2 \Rightarrow E_2 \mid E \mid v \mid E \mid E \tau \mid \text{pack } \tau_1, E \text{ as } \exists \alpha. \tau \mid \text{unpack } E \text{ as } \alpha, x \text{ in } E_2 \mid \text{roll } E \mid \text{unroll } E \mid \text{ref } E \mid !E \mid E ::= e \mid v ::= E \mid E == e \mid v == E \]

1.2 Dynamic Semantics

![semantics_diagram]
\[
(h; e) \xrightarrow{I} (h'; e')
\]

- \(l \notin \text{dom}(h)\)
  
\[
(h; E[\text{ref } v]) \xrightarrow{I} (h[l \mapsto v]; E[l])
\]

- \(h(l) = v\)
  
\[
(h; E[l]) \xrightarrow{I} (h; E[v])
\]

- \(l \in \text{dom}(h)\)
  
\[
(h; E[l := v]) \xrightarrow{I} (h[l \mapsto v]; E[\emptyset])
\]

\[
(h; e) \xrightarrow{I}^n (h'; e')
\]

\[
(h; e) \xrightarrow{e}^0 (h; e)
\]

\[
(h; e) \xrightarrow{I}^n (h''; e'') \quad (h''; e'') \xrightarrow{I}^m (h'; e')
\]

\[
(h; e) \xrightarrow{I}^n (h'; e''') \quad e'''' \sim^m e'
\]

\[
(h; e) \xrightarrow{n+1} (h'; e')
\]

\[
(h; e) \xrightarrow{n+m} (h'; e')
\]

\[
(h; e) \Downarrow^n (h'; e') \quad \overset{\text{def}}{=} \exists (h; e) \xrightarrow{I}^n (h'; e') \land
\]

\[
\exists k, h'', e'', P^k e'' \sim e'' \lor (h'; e') \xrightarrow{\bar{P}} (h''; e'')
\]

\[
(h; e) \Downarrow^n \quad \overset{\text{def}}{=} \exists h', e'. (h; e) \Downarrow^n (h'; e')
\]

\[
(h; e) \Downarrow \quad \overset{\text{def}}{=} \exists n. (h; e) \Downarrow^n
\]
1.3 Static Semantics

Variable Context $\Gamma ::= \cdot | \Gamma, \alpha | \Gamma, x : \tau$

Heap Context $\Sigma ::= \cdot | \Sigma, l : \tau$

$\Gamma \vdash \cdot$

$\vdash \Gamma, \alpha$

$\vdash \Gamma, x : \tau$

$\Gamma \vdash \tau$

$\alpha \in \Gamma$

$\Gamma \vdash \alpha$

$\Gamma \vdash \unit$

$\Gamma \vdash \int$

$\Gamma \vdash \bool$

$\Gamma \vdash \ref \tau$

$\Gamma \vdash \tau_1 \times \tau_2$

$\Gamma \vdash \tau_1 + \tau_2$

$\Gamma \vdash \tau_1 \rightarrow \tau_2$

$\Gamma \vdash \forall \alpha. \tau$

$\Gamma \vdash \exists \alpha. \tau$

$\Gamma \vdash \mu \alpha. \tau$

$\Gamma; \Sigma \vdash e : \tau$

$\Gamma(x) = \tau$

$\Gamma; \Sigma \vdash x : \tau$

$\Gamma; \Sigma \vdash \emptyset : \unit$

$\Gamma; \Sigma \vdash n : \int$

$\Gamma; \Sigma \vdash \true : \bool$

$\Gamma; \Sigma \vdash \false : \bool$

$\Gamma; \Sigma \vdash \if e \then e_1 \else e_2 : \tau$

$\Gamma; \Sigma \vdash e_1 : \tau_1$

$\Gamma; \Sigma \vdash (e_1, e_2) : \tau_1 \times \tau_2$

$\Gamma; \Sigma \vdash e : \tau_1 \times \tau_2$

$\Gamma; \Sigma \vdash \fst e : \tau_1$

$\Gamma; \Sigma \vdash \snd e : \tau_2$

$\Gamma; \Sigma \vdash \inl e : \tau_1 + \tau_2$

$\Gamma; \Sigma \vdash \inr e : \tau_1 + \tau_2$

$\Gamma; \Sigma \vdash \case e \of \inl e_1 \Rightarrow e_1 | \inr e_2 \Rightarrow e_2 : \tau$

$\Gamma; \Sigma \vdash \lambda x : \tau_1. e : \tau_2$

$\Gamma; \Sigma \vdash e_1 : \tau_2 \rightarrow \tau$

$\Gamma; \Sigma \vdash e_2 : \tau_2$

$\Gamma; \Sigma \vdash \case e \of \inl e_1 \Rightarrow e_1 | \inr e_2 \Rightarrow e_2 : \tau$

$\Gamma; \Sigma \vdash \forall \alpha. e : \forall \alpha. \tau$

$\Gamma; \Sigma \vdash e_1 : \forall \alpha. \tau$

$\Gamma; \Sigma \vdash \pi_{\alpha[\tau_1]} e : \forall \alpha. \tau$

$\Gamma; \Sigma \vdash \alpha.e : \forall \alpha. \tau$

$\Gamma; \Sigma \vdash \pack \tau_1, e \alpha \tau : \exists \alpha. \tau$

$\Gamma; \Sigma \vdash \unpack e_1 \alpha \tau : \exists \alpha. \tau$

$\Gamma; \Sigma \vdash e : \mu \alpha. \tau$

$\Gamma; \Sigma \vdash \roll e : \mu \alpha. \tau$

$\Gamma; \Sigma \vdash e_1 : \alpha.\tau$

$\Gamma; \Sigma \vdash \unroll e : [\mu \alpha. \tau][\alpha \tau]$
1.4 Contexts and Contextual Equivalence

\[ \vdash h \vdash h : \Sigma \]

\[ \text{FV}(h) = \emptyset \quad \text{locs}(h) \subseteq \text{dom}(h) \]

\[ \forall l \in \text{dom}(\Sigma). \vdash h(l) : \Sigma(l) \]

\[ \vdash h : \Sigma \]

Contexts \( C \) ::= \([\_] | \ o(e_1, \ldots, e_{i-1}, C, e_{i+1}, \ldots, e_n) | \ \text{if} \ C \ \text{then} \ e_1 \ \text{else} \ e_2 | \ \text{if} \ e \ \text{then} \ C \ \text{else} \ e_2 | \ \text{case} \ C \ \text{of} \ \text{inl} \ x_1 \Rightarrow e_1 | \ \text{inr} \ x_2 \Rightarrow e_2 | \ \text{case} \ e \ \text{of} \ \text{inl} \ x_1 \Rightarrow C | \ \text{inr} \ x_2 \Rightarrow e_2 | \ \text{case} \ e \ \text{of} \ \text{inl} \ x_1 \Rightarrow e_1 | \ \text{inr} \ x_2 \Rightarrow C | \ \lambda x : \tau. C | \ C \ e | \ e \ C | \ \Lambda \alpha. C | \ C \ \tau | \ \text{pack} \ \tau_1, C \ \text{as} \ \exists \alpha. \ \tau | \ \text{unpack} \ C \ \text{as} \ \alpha, x \ \text{in} \ e_2 | \ \text{unpack} \ e_1 \ \text{as} \ \alpha, x \ \text{in} \ C | \ \text{roll} \ C | \ \text{unroll} \ C | \ \text{ref} \ C | \ !C | \ C ::= e | \ e ::= C | \ C == e | \ e =e = C \]

\[ \vdash C : (\Gamma; \Sigma \vdash \tau) \rightsquigarrow (\Gamma'; \Sigma' \vdash \tau') \]

\[ \Gamma \subseteq \Gamma' \quad \Sigma \subseteq \Sigma' \]

\[ \vdash [\_] : (\Gamma; \Sigma \vdash \tau) \rightsquigarrow (\Gamma'; \Sigma' \vdash \tau) \]

\[ \vdash C : (\Gamma; \Sigma \vdash \tau) \rightsquigarrow (\Gamma'; \Sigma' \vdash \text{bool}) \quad \Gamma'; \Sigma' \vdash e_1 : \tau' \quad \Gamma'; \Sigma' \vdash e_2 : \tau' \]

\[ \vdash \text{if} \ C \ \text{then} \ e_1 \ \text{else} \ e_2 : (\Gamma; \Sigma \vdash \tau) \rightsquigarrow (\Gamma'; \Sigma' \vdash \tau') \]

\[ \vdash C : (\Gamma; \Sigma \vdash \tau) \rightsquigarrow (\Gamma'; \Sigma' \vdash \tau') \quad \vdash C : (\Gamma; \Sigma \vdash \tau) \rightsquigarrow (\Gamma'; \Sigma' \vdash \tau') \]

\[ \vdash \text{if} \ e \ \text{then} \ C' \ \text{else} \ C \ : (\Gamma; \Sigma \vdash \tau) \rightsquigarrow (\Gamma'; \Sigma' \vdash \tau') \]

\[ \vdash C : (\Gamma; \Sigma \vdash \tau) \rightsquigarrow (\Gamma'; \Sigma' \vdash \tau_1) \quad \Gamma' \vdash \text{snd} : \tau_1 \times \tau_2 \]

\[ \vdash \text{fst} : (\Gamma; \Sigma \vdash \tau) \rightsquigarrow (\Gamma'; \Sigma' \vdash \tau_1) \]

\[ \vdash \text{inr} : (\Gamma; \Sigma \vdash \tau) \rightsquigarrow (\Gamma'; \Sigma' \vdash \tau_1) \]

\[ \vdash \text{inl} : (\Gamma; \Sigma \vdash \tau) \rightsquigarrow (\Gamma'; \Sigma' \vdash \tau_1 + \tau_2) \]

\[ \vdash \text{case} \ C \ \text{of} \ \text{inl} \ x_1 \Rightarrow e_1 | \ \text{inr} \ x_2 \Rightarrow e_2 : (\Gamma; \Sigma \vdash \tau) \rightsquigarrow (\Gamma'; \Sigma' \vdash \tau') \]

\[ \vdash \text{case} \ e \ \text{of} \ \text{inl} \ x_1 \Rightarrow C | \ \text{inr} \ x_2 \Rightarrow e_2 : (\Gamma; \Sigma \vdash \tau) \rightsquigarrow (\Gamma'; \Sigma' \vdash \tau') \]

\[ \vdash \text{case} \ e \ \text{of} \ \text{inl} \ x_1 \Rightarrow e_1 | \ \text{inr} \ x_2 \Rightarrow e_2 : (\Gamma; \Sigma \vdash \tau) \rightsquigarrow (\Gamma'; \Sigma' \vdash \tau') \]
Definition 1.1 (Contextual Equivalence)

\[ \Gamma ; \Sigma \vdash e_1 : \tau \quad \text{def} \quad \forall C, \Sigma', \tau', h. \quad \vdash C : (\Gamma ; \Sigma \vdash \tau) \Rightarrow (\Gamma'; \Sigma' \vdash \tau') \land \vdash h : \Sigma' \quad \Rightarrow \quad (h; C[e_1]) \downarrow \iff (h; C[e_2]) \downarrow \]
2 LADR

2.1 Well-formedness

Delayed Assertions
\[ H', J' ::= e_1 \leftrightarrow e_2 \mid H' \circ J' \mid H' \lor J' \mid \exists X. H' \mid \Box P' \]

Delayed Formulas
\[ P', Q' ::= e_1 = e_2 \mid e_1 \sim^0 e_2 \mid e_1 \sim^1 e_2 \mid e_1 \sim^* e_2 \mid \top \mid \bot \mid P' \land Q' \mid P' \lor Q' \mid P' \Rightarrow Q' \mid \forall X. P' \mid \exists X. P' \mid \forall R. P' \mid \exists R. P' \mid \tau \in P' \mid \mathbf{a} \mid \mathbf{p}. P' \mid \mathbf{Val} \mid \mathbf{Const} \tau b \mid \mathbf{Loc} \mid \mathbf{Term} i \mid \mathbf{\Delta} P' \]

\[ \vdash X \text{ ok} \]

\[ \vdash \begin{array}{ll}
X \vdash x \notin X & \vdash X, x \text{ ok} \\
X \vdash \alpha \notin X & \vdash X, \alpha \text{ ok}
\end{array} \]

\[ \vdash X \text{ term} \]

\[ \vdash \begin{array}{ll}
\text{FV}(e) \subseteq X & X \vdash e \text{ term}
\end{array} \]

\[ \vdash R \text{ ok} \]

\[ \vdash \begin{array}{ll}
\vdash R, a \notin R & \vdash R, p \notin R
\end{array} \]

\[ \vdash X; R \vdash L \text{ ok} \]

\[ \vdash \begin{array}{ll}
p \in R & p \notin \text{dom}(L) \quad \text{arity}(p) = \text{arity}(a) \quad \vdash R, a \text{ ok} \\
X; R, a \vdash A : \text{Rel}(0) & X; R, a \vdash H : \text{Asrt} \quad H \text{ delayed}
\end{array} \]

\[ X; R \vdash \Delta, p \times a.(A, H) \text{ ok} \]

\[ \vdash X; R \vdash P \text{ ok} \]

\[ \vdash \begin{array}{ll}
X; R \vdash P_1 : \text{Rel}(0) & \cdots X; R \vdash P_n : \text{Rel}(0)
\end{array} \]

\[ X; R \vdash P_1, \ldots, P_n \text{ ok} \]

\[ \vdash X; R; L; P \text{ ok} \]

\[ \vdash \begin{array}{ll}
X \vdash x \notin X & \vdash X, x \text{ ok}
\end{array} \]

\[ \vdash \begin{array}{ll}
X; R \vdash \Delta & X; R \vdash P \text{ ok}
\end{array} \]

\[ \vdash X; R \vdash \Delta, L \times P \text{ ok} \]

\[ \vdash X; R \vdash P : \text{Rel}(n) \]

\[ \vdash \begin{array}{ll}
X \vdash e_1 \text{ term} & X \vdash e_2 \text{ term}
\end{array} \]

\[ \vdash \begin{array}{ll}
X; R \vdash e_1 \sim^* e_2 : \text{Rel}(0) & X; R \vdash e_1 \sim^0 e_2 : \text{Rel}(0)
\end{array} \]

\[ \vdash \begin{array}{ll}
X; R \vdash e_1 \sim^1 e_2 : \text{Rel}(0) & X; R \vdash e_1 \sim^1 e_2 : \text{Rel}(0)
\end{array} \]

\[ \vdash \begin{array}{ll}
X; R \vdash \top : \text{Rel}(0) & X; R \vdash \bot : \text{Rel}(0)
\end{array} \]

\[ \vdash \begin{array}{ll}
X; R \vdash P : \text{Rel}(0) & X; R \vdash Q : \text{Rel}(0)
\end{array} \]

\[ \vdash X; R \vdash P \land Q : \text{Rel}(0) \]

\[ \vdash X; R \vdash P \lor Q : \text{Rel}(0) \]
2.2 Additional Inference Rules

\[ \begin{array}{llll}
C \vdash ⊤ & \quad & C \vdash \neg \top & \quad & C \vdash \neg \neg \top & \quad & C \vdash \exists \forall \cdot P & \quad & C, R, P \vdash \neg \top \\
C \vdash \neg \top & \quad & \frac{C \vdash P \lor Q}{C \vdash \neg \top} & \quad & C, P \vdash \neg \top & \quad & \frac{C, Q \vdash \neg \top}{C \vdash \neg \top} & \quad & \frac{C \vdash \exists \forall \cdot P}{C, R, P \vdash \neg \top} & \quad & \frac{C \vdash \forall \exists \cdot P}{C \vdash \exists \forall \cdot P}
\end{array} \]

\[ \begin{array}{llll}
C \vdash P & \quad & C \vdash \top & \quad & \frac{C \vdash P \land Q}{C \vdash P} & \quad & \frac{C \vdash P \land Q}{C \vdash Q} & \quad & \frac{C \vdash P \lor Q}{C \vdash P \land Q} & \quad & \frac{C \vdash P \lor Q}{C \vdash Q}
\end{array} \]

\[ \begin{array}{llll}
C \vdash Q & \quad & \frac{C \vdash P \rightarrow Q}{C \vdash P \lor Q} & \quad & \frac{C \vdash P \rightarrow Q}{C \vdash P} & \quad & \frac{C \vdash Q}{C \vdash P \rightarrow Q} & \quad & \frac{C \vdash Q}{C \vdash P \rightarrow Q} & \quad & \frac{C \vdash Q}{C \vdash P \rightarrow Q}
\end{array} \]

\[ \begin{array}{llll}
C, X \vdash P & \quad & \frac{C \vdash \forall \forall \cdot P}{C \vdash \forall \forall \cdot P} & \quad & \frac{C \vdash \forall \forall \cdot P}{C \vdash \forall \forall \cdot P} & \quad & \frac{C \vdash \forall \forall \cdot P}{C \vdash \forall \forall \cdot P} & \quad & \frac{C \vdash \forall \forall \cdot P}{C \vdash \forall \forall \cdot P}
\end{array} \]

\[ \begin{array}{llll}
C \vdash H \Rightarrow H' & \quad & \frac{C \vdash H \Rightarrow H'}{C \vdash H \Rightarrow H'} & \quad & \frac{C \vdash H \Rightarrow H'}{C \vdash H \Rightarrow H'} & \quad & \frac{C \vdash H \Rightarrow H'}{C \vdash H \Rightarrow H'} & \quad & \frac{C \vdash H \Rightarrow H'}{C \vdash H \Rightarrow H'}
\end{array} \]

\[ \begin{array}{llll}
C \vdash H \Rightarrow H' & \quad & \frac{C \vdash H \Rightarrow H'}{C \vdash H \Rightarrow H'} & \quad & \frac{C \vdash H \Rightarrow H'}{C \vdash H \Rightarrow H'} & \quad & \frac{C \vdash H \Rightarrow H'}{C \vdash H \Rightarrow H'} & \quad & \frac{C \vdash H \Rightarrow H'}{C \vdash H \Rightarrow H'}
\end{array} \]

\[ \begin{array}{llll}
C \vdash H \Rightarrow H' & \quad & \frac{C \vdash H \Rightarrow H'}{C \vdash H \Rightarrow H'} & \quad & \frac{C \vdash H \Rightarrow H'}{C \vdash H \Rightarrow H'} & \quad & \frac{C \vdash H \Rightarrow H'}{C \vdash H \Rightarrow H'} & \quad & \frac{C \vdash H \Rightarrow H'}{C \vdash H \Rightarrow H'}
\end{array} \]

\[ \begin{array}{llll}
C \vdash H \Rightarrow H' & \quad & \frac{C \vdash H \Rightarrow H'}{C \vdash H \Rightarrow H'} & \quad & \frac{C \vdash H \Rightarrow H'}{C \vdash H \Rightarrow H'} & \quad & \frac{C \vdash H \Rightarrow H'}{C \vdash H \Rightarrow H'} & \quad & \frac{C \vdash H \Rightarrow H'}{C \vdash H \Rightarrow H'}
\end{array} \]
2.3 Model

\[ |A|_{\delta} \]

\[ |e_1 = e_2|_{\delta} \overset{\text{def}}{=} e_1 = e_2 \]
\[ |\top|_{\delta} \overset{\text{def}}{=} \top \]
\[ |\bot|_{\delta} \overset{\text{def}}{=} \bot \]
\[ |A \land B|_{\delta} \overset{\text{def}}{=} |A|_{\delta} \land |B|_{\delta} \]
\[ |A \lor B|_{\delta} \overset{\text{def}}{=} |A|_{\delta} \lor |B|_{\delta} \]
\[ |A \Rightarrow B|_{\delta} \overset{\text{def}}{=} |A|_{\delta} \Rightarrow |B|_{\delta} \]
\[ |\forall \mathcal{X}.A|_{\delta} \overset{\text{def}}{=} \forall \gamma \in [\mathcal{X}]. |\gamma A|_{\delta} \]
\[ |\exists \mathcal{X}.A|_{\delta} \overset{\text{def}}{=} \exists \gamma \in [\mathcal{X}]. |\gamma A|_{\delta} \]
\[ |\forall R.A|_{\delta} \overset{\text{def}}{=} \forall \delta' \in [R]. |A|_{\delta, \delta'} \]
\[ |\exists R.A|_{\delta} \overset{\text{def}}{=} \exists \delta' \in [R]. |A|_{\delta, \delta'} \]
\[ |\tau \in A|_{\delta} \overset{\text{def}}{=} |A|_{\delta\tau} \]
\[ |a|_{\delta\tau} \overset{\text{def}}{=} \delta(a)\tau \]
\[ |\tau.A|_{\delta\tau} \overset{\text{def}}{=} |A|_{\delta\tau} \]
\[ |Val|_{\delta} \overset{\text{def}}{=} e \text{ value} \]
\[ |Loc|_{\delta} \overset{\text{def}}{=} e \in \text{Loc} \]
2.4 Properties

Lemma 2.1
If $H$ is delayed, then $\|H\|\delta W = \|H\|\delta (\triangleleft W)$.

Lemma 2.2
If $\|P\|\delta W$ and $W' \supseteq W$, then $\|\triangledown P\|\delta W'$.

Lemma 2.3
If $W' \supseteq W$, then $\|H\|\delta W' \supseteq \|H\|\delta W$.

Lemma 2.4
If $W' \vdash (h_1; e_1) \approx (h_2; e_2) : I_R$, $W' \supseteq W$ and $W'.k = W.k$, then $W \vdash (h_1; e_1) \approx (h_2; e_2) : I_R$.

Lemma 2.5 (Substitution)
1. $\|P\|\delta, a\rightarrow A\|\delta = \|P[A/a]\|\delta$
2. $\|H\|\delta, a\rightarrow A\|\delta = \|H[A/a]\|\delta$

Lemma 2.6
If $W.k > 0$, then $\|A\|\delta W = \|A\|\delta$.

Lemma 2.7
If $h_1, h_2 : W$ and $W' \supseteq W$ and $W'.I|_S = W.I|_S$, then $h_1, h_2 : W'$.

Lemma 2.8
1. If $h_1, h_2 : W$, then $h_1, h_2 : \triangledown W$.
2. If $h_1, h_2 : W$, then $h_1, h_2 : \triangleright W$.

Lemma 2.9
If $\|C\|\delta k I$ and $k' \leq k$ and $I' \supseteq I|_{k'}$ and $\text{dom}(I') = \text{dom}(I)$, then $\|C\|\delta k' I'$.

Lemma 2.10
1. If $W' \supseteq W$, then $\triangledown W' \supseteq \triangledown W$.
2. If $W' \supseteq W$, then $\triangleright W' \supseteq \triangleright W$.
3. If $W' \supseteq W$, then $\triangleleft W' \supseteq \triangleleft W$.
4. If $W' \supseteq W$, then $\triangledown W' \supseteq \triangledown W$.

Lemma 2.11
If $\|H\|\delta W(h_1, h_2)$ and $h_1' \supseteq h_1$ and $h_2' \supseteq h_2$, then $\|H\|\delta W(h_1', h_2')$.

Lemma 2.12 (Monotonicity of $\|P\|$ wrt. $\supseteq$)
If $\|P\|\delta W\tau$ and $W' \supseteq W$, then $\|P\|\delta W'\tau$.

Lemma 2.13
If $\|P\|\delta W$, then $\|\triangleleft P\|\delta (\triangledown W)$.
Lemma 2.14

1. If $W' \supseteq \triangleright W$, then there is $\widehat{W}' \supseteq W$ such that $\triangleright \widehat{W}' = W'$.

2. If $W' \supseteq \triangleright W$, then there is $\widehat{W}' \supseteq W$ such that $\triangleright \widehat{W}' = W'$.

**Proof:** In each case define $\widehat{W}' := (W'.k + 1, W'.d, W'.s_1, W'.s_2, I)$, where

\[
\begin{align*}
\text{dom}(I) &= \text{dom}(W'.I), \\
I(\omega) &= W'.I(\omega) \quad \text{if } \omega \notin \text{dom}(W.I), \\
I(\omega) &= (W'.I(\omega).CP, W'.I(\omega).PL, [W.I(\omega).HL]_{W'.k+1}) \quad \text{otherwise}
\end{align*}
\]

□

Lemma 2.15

1. If $W' \supseteq \ltimes \triangleright W$, then there is $\widehat{W}' \supseteq W$ such that $\lhd \triangleright \widehat{W}' = W'$.

2. If $W' \supseteq \ltimes \triangleright W$, then there is $\widehat{W}' \supseteq W$ such that $\lhd \triangleright \widehat{W}' = W'$.

**Proof:** In each case define $\widehat{W}' := (W'.k, W'.d, W'.s_1, W'.s_2, I)$, where

\[
\begin{align*}
\text{dom}(I) &= \text{dom}(W'.I), \\
I(\omega) &= W'.I(\omega) \quad \text{if } \omega \notin \text{dom}(W.I), \\
I(\omega) &= (W'.I(\omega).CP, W'.I(\omega).PL, [W.I(\omega).HL]_{W'.k}) \quad \text{otherwise}
\end{align*}
\]

□
3 Soundness of the Logical Relation

Lemma 3.1 (LR-Value-Monotonicity)
Suppose $C \vdash \nu(\alpha) : \text{Type}$ for all $\alpha \in \text{dom}(\rho)$. If $\text{FV}(\tau) \subseteq \text{dom}(\rho)$, then $C \vdash \nu[\tau] \rho : \text{Type}$.

Lemma 3.2 (Compatibility: Allocation)
If $C \vdash (e_1, e_2) \in \mathcal{E}[\nu \tau \rho]$, then $C \vdash (\text{ref } e_1, \text{ref } e_2) \in \mathcal{E}[\nu \tau \rho]$.

Proof: We show $C_0 \vdash (e_1, e_2) \in \mathcal{E}[\nu \tau \rho] \Rightarrow (\text{ref } e_1, \text{ref } e_2) \in \mathcal{E}[\nu \tau \rho])$, where $C_0 = C.X; C.R; \vdash C.P$. The original claim then follows by rules $\nu$-WEAKEN and $\Rightarrow$-ELIM. Starting with rule $\vdash$-BIND, we need to show $C_1 \vdash (\text{ref } x_1, \text{ref } x_2) \in \mathcal{E}[\nu \tau \rho]$, where $C_1 = \dashv C_0, x_1, x_2, (x_1, x_2) \in \mathcal{V}[\tau] \rho$. Using rule $\vdash$-IMPURE, we enter the separation judgment and are required to show $\{\boxdot \top \} \text{ref } x_1 \equiv \text{ref } x_2 \{\mathcal{V}[\nu \tau \rho]\}$. Since $C_1 \vdash \{\boxdot \top \} \text{ref } x_i \vdash y_i \{y_i \leftarrow x_i\}$ by rule ALLOC, we can apply rule STEP-LR such that it remains to show $C_2 \vdash \{y_i \leftarrow x_1 \ast y_2 \leftarrow x_2\} \equiv y_2 \{\mathcal{V}[\nu \tau \rho]\} \equiv C_2 = \nu C_1, y_1, y_2$. Now, using rule ISL-NEW, we create a new island $p \ast a.(B, H)$ with constant population $A$, where $A = \{(y_1, y_2)\}, B = (\ast \equiv A)$, and $H = \exists x_1, x_2, y_1 \leftarrow x_1 \ast y_2 \leftarrow x_2 \ast \boxdot \varphi(x_1, x_2) \in \mathcal{V}[\tau] \rho$, thus upgrading $C_2$ to $C_3 = C_2, p, p \ast a.(B, H), p \equiv A$. Before we can switch back to the regular judgment using rule ISL-UPD (without actually updating anything), we need to show that $H$ is currently satisfied. This follows easily from the current knowledge about the heap and the assumption $(x_1, x_2) \in \mathcal{V}[\tau] \rho$ together with rule MONO. Back in the regular judgment with $C_4 = \nu C_3, \nu \rho \equiv A$, we must show $(y_1, y_2) \in \mathcal{E}[\nu \tau \rho]$. This follows from rules $\vdash$-RETURN and, after unfolding the definition of $\mathcal{V}[\nu \tau \rho]$, ENTAIL$\leftarrow\vdash$-VAL and $\ast$-INTRO. \hfill $\square$

Lemma 3.3 (Compatibility: Assignment)
If $C \vdash (e_1, e_2) \in \mathcal{E}[\nu \tau \rho]$ and $C \vdash (e_3, e_4) \in \mathcal{E}[\nu \tau \rho]$, then $C \vdash (e_1 := e_3, e_2 := e_4) \in \mathcal{E}[\nu \tau \rho]$. \hfill $\square$

Proof: We show $C_0 \vdash (e_1, e_2) \in \mathcal{E}[\nu \tau \rho] \Rightarrow (e_3, e_4) \in \mathcal{E}[\nu \tau \rho] \Rightarrow (e_1 := e_3, e_2 := e_4) \in \mathcal{E}[\nu \tau \rho]$, where $C_0 = C.X; C.R; \vdash C.P$. The original claim then follows by rules $\nu$-WEAKEN and $\Rightarrow$-ELIM. Starting with rule $\vdash$-BIND, we need to show $C_1 \vdash (x_1 := x_3, x_2 := x_4) \in \mathcal{E}[\nu \tau \rho]$, where $C_1 = \dashv C_0, x_1, x_2, x_3, x_4, (x_1, x_2) \in \mathcal{V}[\nu \tau \rho]$, $(x_3, x_4) \in \mathcal{V}[\nu \tau \rho]$. The assumption about $x_1$ and $x_2$ tells us that $\ast a.(\ldots, H)$, where $H = \exists y_1, y_2, x_1 \leftarrow y_1 \ast x_2 \leftarrow y_2 \ast \boxdot \varphi(y_1, y_2) \in \mathcal{V}[\nu \tau \rho]$. With the help of rule $\ast$-ELIM and the fact that our $C$ is empty, we can extend $C_1$ to $C_2 := C_1, p, p \ast a.(\ldots, H)$ and hence use rule $\vdash$-IMPURE to enter the separation judgment. Here we are required to show $\{H\} x_1 := x_3 \equiv x_2 := x_4 \{\mathcal{V}[\nu \tau \rho]\}$ under $C_3 := C_2, p, p \equiv A$. By combining rules ALLOC and STEP-LR, we need to show $\{H'\} (\langle \rangle \equiv \langle \rangle \{\mathcal{V}[\nu \tau \rho]\}$, where $H' = x_1 \leftarrow x_3 \ast x_2 \leftarrow x_4$. In order to switch back to the regular judgment using rule ISL-UPD, we are required to show that the heap law $H$ is still satisfied. This follows from $H'$ and the assumption $(x_3, x_4) \in \mathcal{V}[\nu \tau \rho]$ together with rule MONO. Consequently, it suffices to show $(\langle \rangle, \langle \rangle) \in \mathcal{E}[\nu \tau \rho]$ under $C_4 := \nu C_3$, which follows by rule $\vdash$-RETURN. \hfill $\square$

Lemma 3.4 (Compatibility: Reference Equality)
If $C \vdash (e_1, e_2) \in \mathcal{E}[\nu \tau \rho]$ and $C \vdash (e_3, e_4) \in \mathcal{E}[\nu \tau \rho]$, then $C \vdash (e_1 \equiv e_3, e_2 \equiv e_4) \in \mathcal{E}[\nu \tau \rho]$. \hfill $\square$

Proof: We show $C_0 \vdash (e_1, e_2) \in \mathcal{E}[\nu \tau \rho] \Rightarrow (e_3, e_4) \in \mathcal{E}[\nu \tau \rho] \Rightarrow (e_1 \equiv e_3, e_2 \equiv e_4) \in \mathcal{E}[\nu \tau \rho]$, where $C_0 = C.X; C.R; \vdash C.P$. The original claim then follows by rules $\nu$-WEAKEN and $\Rightarrow$-ELIM. Starting with rule $\vdash$-BIND, we need to show $C_1 \vdash (x_1 := x_3, x_2 := x_4) \in \mathcal{E}[\nu \tau \rho]$, where $C_1 = \dashv C_0, x_1, x_2, x_3, x_4, (x_1, x_2) \in \mathcal{V}[\nu \tau \rho]$, $(x_3, x_4) \in \mathcal{V}[\nu \tau \rho]$. The assumptions about $x_1, x_2, x_3, x_4$ tell us $\ast a.(B_1, H_1)$ and $\ast a.(B_3, H_3)$, where $B_1 = (a \equiv \{(x_i, x_{i+1})\})$ and $H_1 = \exists y_i, y_{i+1}. x_i \leftarrow$
Let \( y_i \cdot x_{i+1} \equiv 2 y_i \cdot x_{i+1} \in \mathcal{V}[\tau] p \). With the help of rule \( \text{\texttt{\_\_\_elim}} \) and the fact that our \( \mathcal{L} \) is empty, we can extend \( C_1 \) to \( C_2 := C_1, p_1 \cdot p_1 \equiv a.(B_1, H_1) \). Since \( \mathcal{L} \) is now non-empty, another application of that rule requires us to show \( (x_1 = x_3, x_2 = x_4) \in \mathcal{E}[\text{\texttt{\_\_\_bool}}] p \) twice—(1) under \( C_3 := C_2, p_3 \cdot p_3 \equiv a.(B_3, H_3) \) and, separately, (2) under \( C'_3 := C_2, \forall a.B_3 \equiv B_1 \).

For (1) we use rule \( \uparrow\text{-\texttt{\_\_\_impure}} \) to access the knowledge of how the heap looks like with respect to the locations in question. That is, we need to show \( \{H_1 \cdot H_3\} x_1 = x_3 \approx x_2 = x_4 \{\mathcal{E}[\text{\texttt{\_\_\_bool}}] p\} \).

By rule \( \text{\texttt{\_\_\_entail}} \cdot \text{\texttt{\_\_\_sep}} \), \( H_1 \cdot H_3 \) implies \( x_1 \neq x_3 \) as well as \( x_2 \neq x_4 \). Consequently, we know \( x_1 = x_3 \approx \text{false} \) as well as \( x_2 = x_4 \approx \text{false} \). Using rule \( \text{\texttt{\_\_\_expand}} \) and then leaving the separation judgment using rule \( \text{\texttt{\_\_\_isl-upd}} \) (without updating), it suffices to show \( \{\text{false, false}\} \in \mathcal{E}[\text{\texttt{\_\_\_bool}}] p \). This follows by rule \( \uparrow\text{-\texttt{\_\_\_return}} \).

For (2) we can derive \( \{(x_3, x_4)\} = \{(x_1, x_2)\} \) from \( \forall a.B_3 \equiv B_1 \), and thus \( x_3 = x_1 \) and \( x_4 = x_2 \). Consequently, we know \( x_1 = x_3 \approx \text{true} \) as well as \( x_2 = x_4 \approx \text{true} \). All we need from here on are rules \( \uparrow\text{-\texttt{\_\_\_expand}} \) and \( \uparrow\text{-\texttt{\_\_\_return}} \).

**Definition 3.5 (Logical Equivalence Judgment for Values)**

Given \( \Gamma ; \Sigma \vdash v_1 : \tau \) and \( \Gamma ; \Sigma \vdash v_2 : \tau \), we define

\[
\Gamma ; \Sigma \vdash v_1 \equiv_{\text{\texttt{\_\_\_val}}} v_2 : \tau \quad \text{def} \quad \Delta ; \mathcal{R} ; \mathcal{L} ; \mathcal{P} \vdash (\gamma_1 v_1, \gamma_2 v_2) \in \mathcal{V}[\tau] p
\]

where \( \Delta, \mathcal{R}, \mathcal{L}, \mathcal{P}, \gamma_1, \gamma_2, p \) are defined as in the case for terms.

**Lemma 3.6 (Fundamental Property for Values)**

If \( \Gamma ; \Sigma \vdash v : \tau \), then \( \Gamma ; \Sigma \vdash v \equiv_{\text{\texttt{\_\_\_val}}} v : \tau \).

**Definition 3.7 (Wundertüte)**

Suppose \( \Gamma = \alpha, x : \tau \) and \( \Sigma = l : \sigma \).

\[
\text{canonic}(k, d, \Gamma, \Sigma) := (\Delta, \mathcal{R}, \mathcal{L}, \mathcal{P}, \gamma_1, \gamma_2, \rho, W, \delta)\] where

\[
\Delta := \gamma_1, \gamma_2, \pi_1, \pi_2
\]

\[
\mathcal{R} := \tau, \rho
\]

\[
\mathcal{L} := p \equiv a.(a \equiv \{(l, l)\}, \exists y_1, y_2. l \leftarrow_1 y_1 \cdot l \leftarrow_1 y_2 \cdot \square \equiv (y_1, y_2) \in \mathcal{V}[\sigma] )
\]

\[
\mathcal{P} := r : \text{\texttt{\_\_\_Type}}(x_1, x_2) \in \mathcal{V}[\tau] p, l \in \text{\texttt{\_\_\_Val}} 1, l \in \text{\texttt{\_\_\_Val}} 2
\]

\[
\gamma_1 := \alpha \equiv a \cdot x_1, \pi_1 = x_1
\]

\[
\rho := \alpha \equiv a \cdot x_2
\]

\[
W := (k, d, \text{\texttt{\_\_\_dom}}(\Sigma), \text{\texttt{\_\_\_dom}}(\Sigma), I)
\]

\[
\text{\texttt{\_\_\_dom}}(I) := \{\omega_1, \ldots, \omega_n\}
\]

\[
I(\omega_i) := \{(l_i, l_i)\}, \{(l_i, l_i)\}, \lambda CP. \{(W, h_1, h_2) \in \text{\texttt{\_\_\_HeapAtom}} k | (h_1(l_i), h_2(l_i)) \in \mathcal{V}[\sigma] [0W]\}
\]

\[
\delta := \rho \equiv \text{\texttt{\_\_\_pop}}(\omega)
\]

**Lemma 3.8**

If \((\cdot, \mathcal{R}, \mathcal{L}, \mathcal{P}, \emptyset, \emptyset, W, \delta) = \text{\texttt{\_\_\_canonic}}(k, \cdot, \cdot, \cdot, \Sigma)\), then \((\emptyset, \delta, W, W.I, \emptyset) \in \mathcal{V} \vdash \mathcal{R} ; \mathcal{L} ; \mathcal{P} \).
Proof: For this we need to show \([\mathcal{L}]\sigma(W,k)(W,I)\) and \([\mathcal{P}]\sigma W\). The latter is obvious. For the former, we show:

\[
[p_i \alpha a. (a \equiv \{(l_i, l_i)\}); \exists y_1, y_2. l_i \leftarrow y_1 * l_i \leftarrow y_2 * \square p(y_1, y_2) \in \mathcal{V}[\sigma_i]]\delta(W,k)W,I[W_l\{\omega_l\}]
\]

This boils down to the fact that \([\mathcal{L}]((h_1', l_i), (h_2', l_i)) \in \mathcal{V}[\sigma_i]]\delta\omega W' is equivalent to

\[
[\exists y_1, y_2. l_i \leftarrow y_1 * l_i \leftarrow y_2 * \square p(y_1, y_2) \in \mathcal{V}[\sigma_i]]\delta(h_1', h_2').
\]

Lemma 3.9 (Heap Parametricity)
If \(\vdash h : \Sigma\) and \((, R, L, P, \emptyset, \emptyset, W, \delta) = \text{canonic}(k, d, \cdot, \Sigma)\), then \(h, h : W\).

Proof: This boils down to showing \(h, h : _{\text{dom}(W,I)} W\). If \(k = 0\), there is nothing to show. Otherwise, let \(h^1 = h|\{l_1\}\), where we suppose \(\Sigma = \overline{I : \sigma}\). We claim \((\sigma W, h^1, h^1) \in W.I(\omega_1).HL(W.I(\omega_1).PL)\), i.e., \((\sigma W, h^1, h^1) \in \text{HeapAtom}_W\) and \([\mathcal{L}(h(l_i), h(l_i)) \in \mathcal{V}[\sigma_i]]\delta\omega W\). The former is obvious. For the latter, note that \(\vdash h(l_i) : \Sigma(l_i)\), for which the Fundamental Property for Values yields \(\vdash R; L; P \vdash (h(l_i), h(l_i)) \in \mathcal{V}[\sigma_i]\). By Lemmas 3.8 and 2.12 we are done.

Lemma 3.10 (Adequacy)
If \(\vdash \Sigma \vdash e_1 \approx^\text{log} e_2 : \tau\) and \(\vdash \Sigma\), then \((h; e_1) \Downarrow\) iff \((h; e_2) \Downarrow\).

Proof: Suppose \((h; e_1) \Downarrow\) (the other direction is symmetric). Let \((, R, L, P, \emptyset, \emptyset, W, \delta) = \text{canonic}(j + 1, \rightarrow, \Sigma)\). By unfolding the definition of \(\approx^\text{log}\) we know \(\vdash R; L; P \vdash (e_1, e_2) \in \mathcal{E}[\tau]\). By Lemma 3.8, \([e_1, e_2] \in \mathcal{E}[\tau]\delta W\). Since \(h, h : W\) by Heap Parametricity, this yields \(W \vdash (h; e_1) \approx (h; e_2) : \mathcal{V}[\tau]\delta\). Finally, since \(j < W.k\), we learn \((h; e_2) \Downarrow\).

Theorem 3.11 (Soundness w.r.t. Contextual Equivalence)
If \(\Gamma; \Sigma \vdash e_1 \approx^\text{ctx} e_2 : \tau\), then \(\Gamma; \Sigma \vdash e_1 \approx^\text{ctx} e_2 : \tau\).

Proof: Suppose \(\vdash C : (\Gamma; \Sigma \vdash \tau) \rightarrow (\Sigma' \vdash \tau')\) and \(\vdash h : \Sigma'\). By Congruence, \(\vdash \Sigma' \vdash C[e_1] \approx^\text{log} C[e_2] : \tau'\). By Adequacy, \((h; C[e_1]) \Downarrow\) iff \((h; C[e_2]) \Downarrow\).
4 Examples

4.1 Name Generators

Consider:

\[ \tau := \exists \alpha. (\text{unit} \to \alpha) \times (\alpha \times \alpha \to \text{bool}) \]

\[ e_1 := \text{let } x = \text{ref } 0 \text{ in pack int, } (\lambda_\alpha. ++x, \lambda y. \text{fst } y = \text{snd } y) \text{ as } \tau \]

\[ e_2 := \text{pack ref unit, } (\lambda \alpha. \text{ref } \langle \rangle, \lambda y. \text{fst } y == \text{snd } y) \text{ as } \tau \]

To prove these ADTs equivalent, we want to show

\[ \vdash (e_1, e_2) \in E[\tau]. \]

By ↑-IMPURE, we have to show

\[ \vdash \{ \top \} e_1 \approx e_2 \{ V[\tau] \}. \]

By rules step-l, alloc and expand, we need to show

\[ C_1 \vdash \{ x \leftarrow 1 0 \} e'_1 \approx e_2 \{ V[\tau] \} \]

where \( e'_1 \) is the body of the let and \( C_1 = x; \cdots ; x \in \text{Val}_1 \).

Using rule isl-new we introduce an island \( p \) whose population is a partial bijection between the natural numbers that will be generated by \( e'_1 \) and the set of locations allocated by \( e_2 \):

\[ B := \exists n. \exists b \subset \text{Loc}. \text{bij}(a, \{1, \ldots, n\}, b) \]

\[ H := \exists n. (x \leftarrow 1 n) \ast \square(\max(\text{dom}(a), n) \land \text{rng}(a) \subseteq \text{Val}_2) \]

The population law \( B \) states that \( a \) is a bijective relation of the form \( \{1 \mapsto l_1, \ldots, n \mapsto l_n\} \).\(^1\) The heap law \( H \) then verifies that the state of \( x \) always is the maximum \( n \) in the domain of the current population, and that its locations are all valid in the current world. The latter property is important later to show that newly allocated references will always be fresh with respect to the current population. Given \( A = \emptyset \) for the initial population, it is easy to verify that \( C_1 \vdash B[A/a] \).

Consequently, we can now define \( C_2 = C_1, p, p \propto a. (B, H), p \equiv \emptyset \) and need to show

\[ C_2 \vdash \{ x \leftarrow 1 0 \} e'_1 \approx e_2 \{ V[\tau] \} \]

Both expressions are values, so we want to apply rule isl-upd (and ↑-RETURN, using monotonicity of the logical relation) to reduce the problem to

\[ C_3 \vdash (e'_1, e_2) \in V[\tau] \]

where \( C_3 = \dagger C_2, p \equiv \emptyset \). To do so, we first need to prove \( C_2 \vdash B[\emptyset/a] \), which follows as before, and \( C_3 \vdash x \leftarrow 1 0 \Rightarrow H[\emptyset/a] \). The latter consists of eliminating the existential in \( H \) with \( n = 0 \) and then showing both parts of the separating conjunction. The first, \( x \leftarrow 1 0 \), is immediate from the assumption, while the second part is a propositional formula that we can prove separately and then cut into the entailment judgment.

Now, we unroll the definition of \( V[\exists \alpha. \tau] \) and pick

\[ r := (x_1, x_2). x_1 \in \text{Val}_1 \land x_2 \in \text{Val}_2 \land (x_1, x_2) \in p \]

We have to show \( C_3 \vdash r : \text{Type} \) — which follows straightforwardly from rule pop-mono. Now let \( \rho = \alpha \rightarrow r \). By the definition of \( V[\tau' \times \tau''] \rho \), we still have to show

\(1\)The bij predicate as well as the other notation in \( B \) and \( H \) can be defined in our logic.
1. $C_3 \vdash (\lambda x.++x, \lambda \cdot \text{ref} \, () \in \mathcal{V}[\text{unit} \rightarrow \alpha] \rho$

2. $C_3 \vdash (\lambda y. \text{fst} \, y = \text{snd} \, y, \lambda y. \text{fst} \, y = \text{snd} \, y) \in \mathcal{V}[\alpha \times \alpha \rightarrow \text{bool}] \rho$

Let us first turn to (1). We unroll the definition of $\mathcal{V}[\tau' \rightarrow \tau'']$, and apply the introduction rules for $\sqcap, \forall$ and $\Rightarrow$, producing the goal

$$C_4 \vdash ((\lambda x.++x) \, y_1, (\lambda \cdot \text{ref} \, () \, y_2) \in \mathcal{V}[\text{unit} \rightarrow \alpha] \rho$$

where $C_4 = \vdash C_2, y_1, y_2, (y_1, y_2) \in \mathcal{V}[\text{unit}] \rho$ (note that $\vdash C_3 = \vdash C_2$). By $\vdash \text{-EXPAND}, \vdash \text{-IMPURE}$ and $\text{SEP-}\exists$, and expanding the $++$ notation, we have to show the equivalence

$$C_5 \vdash \{ x \leftarrow_1 n * \sqcap P \} (x := !x + 1; !x) \approx \text{ref} \, () \{ \mathcal{V}[\alpha] \rho$$

where $C_5 = C_4, a, n, p \equiv a$ and $P$ is the boxed proposition in $H$. We can step through the left computation using rules $\text{STEP-L}$ and $\text{EXPAND}$ and reach

$$C_5 \vdash \{ x \leftarrow_1 n + 1 * \sqcap P \} n + 1 \approx \text{ref} \, () \{ \mathcal{V}[\alpha] \rho$$

The right side is more tricky, because we have to derive an appropriate freshness property for the new location. We first move $\sqcap P$ to the context (rule $\sqcap \text{-SHIFT}$), producing $C_6 = C_5, \sqcap (\text{max}(\text{dom}(a), n) \land \text{rng}(a) \subseteq \text{Val}_2)$. From there we can derive $C_6 \vdash \text{rng}(a) \subseteq \text{Val}_2$ and then apply rules $\text{ALLOC}$ and $\text{STEP-R}$ to get the goal

$$C_6, y \vdash \{ x \leftarrow_1 n + 1 * y \leftarrow () * x \, \sqcap (y \notin \text{rng}(a)) \, \sqcap P \} n + 1 \approx y \{ \mathcal{V}[\alpha] \rho$$

By $\text{SEP-ENTAIL}$ and $\text{ENTAIL-}\leftarrow \text{-VAL}$, we can further strengthen the heap assertion to include $y \in \text{Loc}$.

Now that the new names have been generated, we update the island with the new population $A = a \cup \{(n + 1, y)\}$, using rule $\text{ISL-UPD}$. To do so, we first have to show $C_6 \vdash p \equiv A$, which is easy given the assumption $p \equiv a$ in the context. Then we have to show that the population law still holds, i.e., $C_6 \vdash \exists n'. \exists b \subseteq \text{Loc}. \, \text{bij}(A, \{1, \ldots, n'\}, b)$. Assuming a suitable set of admissible rules about bijections, this can be derived by picking $n' = n + 1$ and $b = \text{rng}(a) \cup \{y\}$, and using the assumption $\text{max}(\text{dom}(a), n)$.

It also requires the assumptions $y \in \text{Loc}$ and $y \notin \text{rng}(a)$, which we can extract from the heap assertion (rule $\sqcap \text{-SHIFT}$). Likewise, we have to prove that the final heap assertion entails the heap law $H[A/a]$ under the environment $C_7 = \vdash C_6, p \equiv A$. Choosing $n + 1$ for the existential variable $n$ in $H$, the first half, $x \leftarrow_1 n + 1$, follows directly from the heap assertion. The rest can again be derived using only propositional logic.

The final step in this part is to show $C_7 \vdash (n + 1, y) \in \mathcal{V}[\alpha] \rho$. In order to apply rule $\vdash \text{-RETURN}$, a proof for $C_7 \vdash \mathcal{V}[\alpha] \rho$ is required — which we already proved above, modulo weakening.

The rest then is by straightforward propositional reasoning, given the updated population $p \equiv A$ in $C_7$ and our definition of $r$ in terms of $p$.

For part (2) we start as before, yielding the goal

$$C_4' \vdash ((\lambda y. \text{fst} \, y = \text{snd} \, y) \, y_1, (\lambda y. \text{fst} \, y = \text{snd} \, y) \, y_2) \in \mathcal{V}[\text{bool}] \rho$$

with $C_4' = \vdash C_2, y_1, y_2, (y_1, y_2) \in \mathcal{V}[\alpha \times \alpha]$. We can unfold the definitions of $\mathcal{V}[\tau' \times \tau'']$ and $\mathcal{V}[\alpha]$ in the context and eliminate the respective existential, producing

$$C_5' = \vdash C_2, y_1, y_2, y_1', y_2', y_1'' y_2'' y_1 = (y_1', y_1''), y_2 = (y_2', y_2'') \in r, (y_1'', y_2'') \in r.$$
Now \( y_1, y_2 \) in the judgment can be replaced by the respective pairs, and by \( \vdash \text{-EXPAND} \) we are left with
\[
C' \vdash (y_1' = y_1'', y_2' = y_2'') \in \mathcal{V}[[\text{bool}]]\rho.
\]

At this point, the proof essentially boils down to the absolute proposition
\[
\forall a, y_1', y_1'', y_2', y_2''. B \Rightarrow (y_1', y_2') \in a \Rightarrow (y_1'', y_2'') \in a \Rightarrow \\
\exists b \in \{\text{true}, \text{false}\}. (y_1' = y_1'' \sim b) \equiv (y_2' = y_2'' \sim b)
\]
Expanding out the definition of \( B \), this can be proved by straightforward means in the meta logic and thus be assumed as an axiom.

### 4.2 Landin’s Knot

We want to prove that Landin’s Knot – the construction of a fixpoint using backpatching – works. That is, we want to prove the equivalence between the following two expressions of type \( \tau_1 \rightarrow \tau_2 \):
\[
e_1 := \text{let } z = \text{ref } (\lambda x. \bot) \text{ in } (z := (\lambda x. \text{let } f = !y \text{ in } e); !z) \\
e_2 := \text{fix } f(x). e
\]
where \( \text{fix} \) is a standard cbv fixpoint operator, which can be defined in our language as follows:
\[
\text{fix } f(x). e \ := \ \lambda x. (\text{unroll } v) \ x \\
\text{where } v \ := \ \text{roll } \lambda f'. (\lambda f. \lambda x. e)(\lambda x. (\text{unroll } f') \ f' \ x)
\]
We need to show \( \vdash (e_1, e_2) \in E[[\tau_1 \rightarrow \tau_2]], \) or, by rule \( \vdash \text{-IMPURE} \), \( \vdash \{ \square \top \} e_1 \approx e_2 \{ \mathcal{V}[\tau_1 \rightarrow \tau_2] \} \).
By applying \( \text{STEP-L} \) several times, we can reduce this to showing \( \{ z \leftarrow_1 F \} F \approx e_2 \{ \mathcal{V}[\tau_1 \rightarrow \tau_2] \}, \)
where \( F \) is the function value assigned to \( z \) in \( e_1 \).

By rule \( \text{ISL-NEW} \) we introduce an island \( p \) that records the fact that \( y \) will contain \( F \), forever. That is, we choose \( H = z \leftarrow_1 F \) as the heap law. We do not need the population for this proof, so we pick \( B = \top \) and \( A = \emptyset \). By rules \( \text{ISL-UPD} \) and \( \vdash \text{-RETURN} \), it remains to be shown that \( (F, e_2) \in \mathcal{V}[\tau_1 \rightarrow \tau_2] \) under the extended context.

At this point, we invoke rule \( \text{LOB} \) to prove the equivalence of \( F \) and \( e_2 \) under the “coinductive” assumption \( \triangleright(F, e_2) \in \mathcal{V}[\tau_1 \rightarrow \tau_2] \). Unfolding the definition of \( \mathcal{V}[\tau' \rightarrow \tau''] \), we assume \( y_1, y_2 \) with \( (y_1, y_2) \in \mathcal{V}[\tau_1] \) and apply \( \vdash \text{-IMPURE} \) again, such that we now have to show \( \{ z \leftarrow_1 F \} F y_1 \approx e_2 y_2 \{ \mathcal{V}[\tau_2] \} \). Note how our island re-establishes the crucial assertion about \( y \) pointing to \( F \).

We use \( \text{EXPAND} \) to reach the point where both expressions have to make an essential step:
\[
\{ z \leftarrow_1 F \} \ (\text{let } f = !z \text{ in } e[y_1/x]) \approx (\text{unroll } v) \ y_2 \{ \mathcal{V}[\tau_2] \}
\]
Now we can apply \( \text{STEP-LR} \), using \( \text{DEREF} \) on the left and \( \text{UNROLL} \) on the right, yielding, after further reduction
\[
\{ z \leftarrow_1 F \} e[F/f][y_1/x] \approx e[e_2/f][y_2/x] \{ \mathcal{V}[\tau_2] \}
\]
which we have to prove in an earlier world — removing the \( \triangleright \) operator from the assumption \( (F, e_2) \in \mathcal{V}[\tau_1 \rightarrow \tau_2] \) that was introduced by rule \( \text{LOB} \).

We apply \( \text{ISL-UPD} \) once more (boxing and unboxing the monotonous assumptions \( (y_1, y_2) \in \mathcal{V}[\tau_1] \) and \( (F, e_2) \in \mathcal{V}[\tau_1 \rightarrow \tau_2] \) to have them survive) so that the goal becomes \( (e[F/f][y_1/x], e[e_2/f][y_2/x]) \in \mathcal{V}[\tau_2] \). By applying \( \vdash \text{-REDUCE} \) twice this can be expanded to \( ((\lambda f. \lambda x. e) \ F \ y_1, (\lambda f. x e) \ e_2 \ y_2) \in \mathcal{V}[\tau_2] \).
Now use the introduction rules for implication and $\Box$ twice, discharging the assumptions $(y_1, y_2) \in \mathcal{V}[\tau_1]$ and $(F, e_2) \in \mathcal{V}[\tau_1 \rightarrow \tau_2]$. With the definition of $\mathcal{V}[\tau' \rightarrow \tau'']$, this gives

$$(\lambda f. \lambda x. e, \lambda f. \lambda x. e) \in \mathcal{V}[(\tau_1 \rightarrow \tau_2) \rightarrow \tau_1 \rightarrow \tau_2]$$

which holds by the Fundamental Property.