Manifest Contracts for Datatypes
(Supplementary Material)

Taro Sekiyama Yuki Nishida Atsushi Igarashi

{t-sekiym,nishida,igarashi}@fos.kuis.kyoto-u.ac.jp

January 29, 2016

1 Definition

In this section, we formalize our calculus.

1.1 Syntax

The syntax including both programs and run-time terms is given as follows.

Types

T ::= $\mathsf{Bool} \mid x{:}T_1 \to T_2 \mid x{:}T_1 \times T_2 \mid \{x{:}T \mid e\} \mid \tau\langle e\rangle$

Constants, Values, Terms

c ::= $\mathsf{true} \mid \mathsf{false}$

v ::= $c \mid \mathtt{fix}\ f(x{:}T_1){:}T_2 = e \mid \langle T_1 \Leftarrow T_2\rangle^\ell \mid (v_1, v_2) \mid C\langle e\rangle v$

e ::= $c \mid x \mid \mathtt{fix}\ f(x{:}T_1){:}T_2 = e \mid e_1\, e_2 \mid (e_1, e_2) \mid e.1 \mid e.2 \mid C\langle e_1\rangle e_2 \mid \mathsf{match}\, e\, \mathsf{with}\, \overline{C_i\, x_i \to e_i}^{\,i} \mid$
$\mathsf{if}\, e_1\, \mathsf{then}\, e_2\, \mathsf{else}\, e_3 \mid \langle T_1 \Leftarrow T_2\rangle^\ell \mid \Uparrow\ell \mid \langle\{x{:}T \mid e_1\}, e_2, v\rangle^\ell \mid \langle\!\langle\{x{:}T \mid e_1\}, e_2\rangle\!\rangle^\ell$

Datatype definitions

ς ::= $\tau\,\langle x{:}T\rangle = \overline{C_i : T_i}^{\,i} \mid \tau\,\langle x{:}T\rangle = \overline{C_i \parallel D_i : T_i}^{\,i}$

Σ ::= $\varnothing \mid \Sigma, \varsigma$

Evaluation contexts

E ::= $[\,] \mid E\, e_2 \mid v_1\, E \mid (E, e_2) \mid (v_1, E) \mid E.1 \mid E.2 \mid C\langle e_1\rangle E \mid$
$\mathsf{match}\, E\, \mathsf{with}\, \overline{C_i\, x_i \to e_i}^{\,i} \mid \mathsf{if}\, E\, \mathsf{then}\, e_2\, \mathsf{else}\, e_3 \mid$
$\langle\{x{:}T \mid e\}, E, v\rangle^\ell \mid \langle\!\langle\{x{:}T \mid e\}, E\rangle\!\rangle^\ell$

Typing contexts

Γ ::= $\varnothing \mid \Gamma, x{:}T$

Table 1 shows metafunctions to look up information on datatype definitions. Their definitions are omitted since they are straightforward. A type specification, returned by *TypSpecOf* and written $x{:}T_1 \rightarrowtail T_2 \rightarrowtail \tau\langle x\rangle$, of a constructor C consists of the datatype τ that C belongs to, the parameter x of τ and the type T_1 of x, and the argument type T_2 of C. In other words, $\tau = \mathit{TypNameOf}_\Sigma(C)$, $x{:}T_1 = \mathit{ArgTypeOf}_\Sigma(\tau)$ and $T_2 = \mathit{CtrArgOf}_\Sigma(C)$. We omit the type definition environment from these metafunctions for brevity if it is clear from the context.

We use the following familiar notations. We write $\mathsf{FV}(e)$ to denote the set of free variables in a term e, and $e\,\{e'/x\}$ capture avoiding substitution of e' for x in e. We apply similar notations to values and types. We say that a term/value/type is closed if it has no free variables, and identify α-equivalent ones. In addition, we introduce several syntactic sugars. A function type $T_1 \to T_2$ means $x{:}T_1 \to T_2$ where the variable x does not occur free in T_2. We often omit type annotations of $\mathtt{fix}\ f(x{:}T_1){:}T_2 = e$ and write $\lambda\, x{:}T.e$ to denote $\mathtt{fix}\ f(x{:}T) = e$ if f does not occur

$\mathit{TypDefOf}_\Sigma(\tau)$	The definition of τ.
$\mathit{ArgTypeOf}_\Sigma(\tau)$	The parameter of τ and its type.
$\mathit{CtrsOf}_\Sigma(\tau)$	The set of constructors that belong to τ.
$\mathit{TypSpecOf}_\Sigma(C)$	The type specification of C.
$\mathit{TypNameOf}_\Sigma(C)$	The data type that C belongs to.
$\mathit{CtrArgOf}_\Sigma(C)$	The argument type of C.

Table 1: Lookup functions.

in the term e. A let-expression $\mathsf{let}\, x = e_1 \,\mathsf{in}\, e_2$ denotes $(\lambda x{:}T.e_2)\, e_1$ where T is an appropriate type. A datatype τ is said to be monomorphic when the definition of τ does not refer to a type argument variable, and then we write τ to denote an application of τ to a term. Given a binary relation R, the relation R^* denotes the reflexive transitive closure of R.

We define an auxiliary function *unref*, which maps a type to its underlying (non-refinement) type.

$$\begin{array}{lcll} \mathit{unref}(\{x{:}T \mid e\}) & = & \mathit{unref}(T) & \\ \mathit{unref}(T) & = & T & \text{(if } T \text{ is not a refinement type)} \end{array}$$

1.2 Semantics

The semantics of our calculus consists of two relations over closed terms: reduction ($\rightsquigarrow$) and evaluation ($\longrightarrow$). The rules, shown in Figure 1, for these relations rest on a constructor choice function. A constructor choice function δ is a partial function that maps a term of the form $\langle \tau_1\langle e_1\rangle \Leftarrow \tau_2\langle e_2\rangle\rangle^\ell\, C\langle e\rangle v$ to a constructor C_1. We fix Γ and δ through this material and usually omit from relations and judgments.

1.3 Type System

A type system of our calculus consists of three judgments: context well-formedness $\vdash \Gamma$, type well-formedness $\Gamma \vdash T$, and typing $\Gamma \vdash e : T$. The derivation rules for these judgments are shown in Figure 2. The typing rule (T_Conv) mentions type equivalence relation denoted by $\equiv$, which is defined as follows.

Definition 1 (Type Equivalence)**.**

1. *The common subexpression reduction relation $\Rrightarrow$ over types is defined as follows: $T_1 \Rrightarrow T_2$ iff there exist some T, x, e_1 and e_2 such that $T_1 = T\,\{e_1/x\}$ and $T_2 = T\,\{e_2/x\}$ and $e_1 \longrightarrow e_2$.*

2. *The type equivalence $\equiv$ is the symmetric transitive closure of $\Rrightarrow$.*

Next, we define well-formedness of type definition environments and constructor choice functions.

Definition 2 (Well-Formed Type Definition Environments)**.**

1. *Let $\varsigma = \tau\,\langle x{:}T\rangle = \overline{C_i : T_i}^{\,i \in \{1,\ldots,n\}}$. A type definition ς is* well formed *under a type definition environment Σ if it satisfies the followings: (a) $0 < n$. (b) $\Sigma;\varnothing \vdash T$ holds. (c) For any $i \in \{1,\ldots,n\}$, $\Sigma,\varsigma;x{:}T \vdash T_i$ holds.*

2. *Let $\varsigma = \tau\,\langle x{:}T\rangle = \overline{C_i \parallel D_i : T_i}^{\,i \in \{1,\ldots,n\}}$. A type definition ς is* well formed *under a type definition environment Σ if it satisfies the followings: (a) $0 < n$. (b) $\Sigma;\varnothing \vdash T$ holds. (c) For any $i \in \{1,\ldots,n\}$, $\Sigma,\varsigma;x{:}T \vdash T_i$ holds. (d) There exists some datatype τ' in Σ such that constructors $\overline{D_i}^{\,i \in \{1,\ldots,n\}}$ belong to it. (e) For any $i \in \{1,\ldots,n\}$, T_i is compatible with the argument type of D_i under Σ,ς, that is, $\Sigma,\varsigma \vdash T_i \parallel \mathit{CtrArgOf}_\Sigma(D_i)$ holds.*

3. *A type definition environment Σ is* well formed *if for any Σ_1, ς and Σ_2, $\Sigma = \Sigma_1,\varsigma,\Sigma_2$ implies that ς is well formed under Σ_1. We write $\vdash \Sigma$ to denote that Σ is well formed.*

$\boxed{e_1 \rightsquigarrow e_2}$ **Reduction Rules**

$$
\begin{array}{rcll}
(\texttt{fix}\ f(x{:}T_1){:}T_2 = e)\, v & \rightsquigarrow & e\,\{v/x, \texttt{fix}\ f(x{:}T_1){:}T_2 = e/f\} & \text{(R\_Beta)}\\
(v_1,v_2).1 \rightsquigarrow v_1 \quad \text{(R\_Proj1)} & & \mathsf{if\,true\,then}\, e_1\, \mathsf{else}\, e_2 \rightsquigarrow e_1 & \text{(R\_IfTrue)}\\
(v_1,v_2).2 \rightsquigarrow v_2 \quad \text{(R\_Proj2)} & & \mathsf{if\,false\,then}\, e_1\, \mathsf{else}\, e_2 \rightsquigarrow e_2 & \text{(R\_IfFalse)}\\
\mathsf{match}\, C_j\langle e\rangle v\, \mathsf{with}\, \overline{C_i\, x_i \to e_i}^{\,i} & \rightsquigarrow & e_j\,\{v/x_j\} \qquad (\text{where } C_j \in \overline{C_i}^{\,i}) & \text{(R\_Match)}\\
\langle \mathsf{Bool} \Leftarrow \mathsf{Bool}\rangle^{\ell}\, v & \rightsquigarrow & v & \text{(R\_Base)}\\
\langle x{:}T_{11} \to T_{12} \Leftarrow x{:}T_{21} \to T_{22}\rangle^{\ell}\, v & \rightsquigarrow & (\lambda x{:}T_{11}.\mathsf{let}\, y = \langle T_{21} \Leftarrow T_{11}\rangle^{\ell}\, x\, \mathsf{in}\, \langle T_{12} \Leftarrow T_{22}\,\{y/x\}\rangle^{\ell}\,(v\,y)) & \\
& & (\text{where } y \text{ is fresh}) & \text{(R\_Fun)}\\
\langle x{:}T_{11} \times T_{12} \Leftarrow x{:}T_{21} \times T_{22}\rangle^{\ell}\,(v_1,v_2) & \rightsquigarrow & \mathsf{let}\, x = \langle T_{11} \Leftarrow T_{21}\rangle^{\ell}\, v_1\, \mathsf{in}\, (x, \langle T_{12} \Leftarrow T_{22}\,\{v_1/x\}\rangle^{\ell}\, v_2) & \text{(R\_Prod)}\\
\langle T_1 \Leftarrow \{x{:}T_2 \mid e\}\rangle^{\ell}\, v & \rightsquigarrow & \langle T_1 \Leftarrow T_2\rangle^{\ell}\, v & \text{(R\_Forget)}\\
\langle \{x{:}T_1 \mid e\} \Leftarrow T_2\rangle^{\ell}\, v & \rightsquigarrow & \langle\!\langle \{x{:}T_1 \mid e\}, \langle T_1 \Leftarrow T_2\rangle^{\ell}\, v\rangle\!\rangle^{\ell} & \text{(R\_PreCheck)}\\
& & (\text{where } T_2 \text{ is not a refinement type}) & \\
\langle \tau_1\langle e_1\rangle \Leftarrow \tau_2\langle e_2\rangle\rangle^{\ell}\, C_2\langle e\rangle v & \rightsquigarrow & C_1\langle e_1\rangle(\langle T_1'\,\{e_1/x_1\} \Leftarrow T_2'\,\{e_2/x_2\}\rangle^{\ell}\, v) & \text{(R\_Datatype)}\\
\multicolumn{4}{r}{(\text{where } \tau_1 \neq \tau_2 \text{ or } \tau_1 \text{ is not monomorphic, and } \delta(\langle \tau_1\langle e_1\rangle \Leftarrow \tau_2\langle e_2\rangle\rangle^{\ell}\, C_2\langle e\rangle v) = C_1 \text{ and}}\\
\multicolumn{4}{r}{\mathit{ArgTypeOf}(\tau_i) = x_i{:}T_i \text{ and } \mathit{CtrArgOf}(C_i) = T_i' \text{ for } i \in \{1,2\}\ )}\\
\langle \tau \Leftarrow \tau\rangle^{\ell}\, v & \rightsquigarrow & v & \text{(R\_DatatypeMono)}\\
\langle \tau_1\langle e_1\rangle \Leftarrow \tau_2\langle e_2\rangle\rangle^{\ell}\, v & \rightsquigarrow & \Uparrow\ell & \text{(R\_DatatypeFail)}\\
\multicolumn{4}{r}{(\text{where } \tau_1 \neq \tau_2 \text{ or } \tau_1 \text{ is not monomorphic, and } \delta(\langle \tau_1\langle e_1\rangle \Leftarrow \tau_2\langle e_2\rangle\rangle^{\ell}\, v) \text{ is undefined }\ )}\\
\langle\!\langle \{x{:}T \mid e\}, v\rangle\!\rangle^{\ell} & \rightsquigarrow & \langle \{x{:}T \mid e\}, e\,\{v/x\}, v\rangle^{\ell} & \text{(R\_Check)}\\
\langle \{x{:}T \mid e\}, \mathsf{true}, v\rangle^{\ell} \rightsquigarrow v \quad \text{(R\_OK)} & & \langle \{x{:}T \mid e\}, \mathsf{false}, v\rangle^{\ell} \rightsquigarrow \Uparrow\ell & \text{(R\_Fail)}
\end{array}
$$

$\boxed{e_1 \longrightarrow e_2}$ **Evaluation Rules**

$$
\frac{e_1 \rightsquigarrow e_2}{E[e_1] \longrightarrow E[e_2]}\ \text{E\_Red}
\qquad\qquad
\frac{E \neq [\,]}{E[\Uparrow\ell] \longrightarrow \Uparrow\ell}\ \text{E\_Blame}
$$

Figure 1: Semantics.

Definition 3 (Compatible Constructors)**.** *The compatibility relation $\parallel$ over constructors is the least equivalence relation satisfying the following rule.*

$$
\frac{\mathit{TypNameOf}(C_i) = \tau \qquad \mathit{TypDefOf}(\tau) = \mathsf{type}\,\tau\,\langle y{:}T\rangle = \overline{C_j \parallel D_j \,:\, T_j}^{\,j}}{C_i \parallel D_i}
$$

The function CompatCtrsOf, which maps a datatype τ and a constructor C to the set of compatible constructors of τ, is defined as follows:

$$\mathit{CompatCtrsOf}(\tau, C) = \{D \mid C \parallel D \ \mathsf{and}\ \mathit{TypNameOf}(D) = \tau\}.$$

Definition 4 (Term Equivalence)**.**

1. *The common subexpression reduction relation $\Rrightarrow$ over terms is defined as follows: $e_1 \Rrightarrow e_2$ iff there exist some e, x, e_1' and e_2' such that $e_1 = e\,\{e_1'/x\}$ and $e_2 = e\,\{e_2'/x\}$ and $e_1' \longrightarrow e_2'$.*

2. *The term equivalence $\equiv$ is the symmetric transitive closure of $\Rrightarrow$.*

Definition 5 (Well-Formed Constructor Choice Functions)**.** *A constructor choice function δ is* well formed *iff*

1. *if $C_1 = \delta(\langle \tau_1\langle e_1\rangle \Leftarrow \tau_2\langle e_2\rangle\rangle^{\ell}\, C_2\langle e\rangle v)$, then $C_1 \in \mathit{CompatCtrsOf}(\tau_1, C_2)$; and*

2. *for any e_1, e_2 and C, if $e_1 \equiv e_2$ and $\delta(e_1) = C$, then $\delta(e_2) = C$.*

Finally, we use notation $\Rrightarrow^i$ to denote i-times composition of $\Rrightarrow$.

$\boxed{\vdash \Gamma}$ **Typing Context Well-Formedness Rules**

$$\frac{}{\vdash \varnothing}\ \textsc{WC\_Empty} \qquad \frac{\vdash \Gamma \quad \Gamma \vdash T}{\vdash \Gamma, x{:}T}\ \textsc{WC\_ExtendVar}$$

$\boxed{\Gamma \vdash T}$ **Type Well-Formedness Rules**

$$\frac{\vdash \Gamma}{\Gamma \vdash \mathsf{Bool}}\ \textsc{WT\_Base} \qquad \frac{\Gamma \vdash T_1 \quad \Gamma, x{:}T_1 \vdash T_2}{\Gamma \vdash x : T_1 \to T_2}\ \textsc{WT\_Fun} \qquad \frac{\Gamma \vdash T_1 \quad \Gamma, x{:}T_1 \vdash T_2}{\Gamma \vdash x{:}T_1 \times T_2}\ \textsc{WT\_Prod}$$

$$\frac{\Gamma \vdash T \quad \Gamma, x{:}T \vdash e : \mathsf{Bool}}{\Gamma \vdash \{x{:}T \mid e\}}\ \textsc{WT\_Refine} \qquad \frac{\mathit{ArgTypeOf}(\tau) = x{:}T \quad \Gamma \vdash e : T}{\Gamma \vdash \tau\langle e\rangle}\ \textsc{WT\_Datatype}$$

$\boxed{\Gamma \vdash e : T}$ **Typing Rules**

$$\frac{\vdash \Gamma \quad c \in \{\mathsf{true}, \mathsf{false}\}}{\Gamma \vdash c : \mathsf{Bool}}\ \textsc{T\_Const} \qquad \frac{\vdash \Gamma \quad x{:}T \in \Gamma}{\Gamma \vdash x : T}\ \textsc{T\_Var} \qquad \frac{\vdash \Gamma \quad \varnothing \vdash T}{\Gamma \vdash \Uparrow\ell : T}\ \textsc{T\_Blame}$$

$$\frac{\Gamma, f{:}(x{:}T_1 \to T_2), x{:}T_1 \vdash e : T_2 \quad f \notin \mathsf{FV}(T_2)}{\Gamma \vdash \mathtt{fix}\ f(x{:}T_1){:}T_2 = e : x{:}T_1 \to T_2}\ \textsc{T\_Abs} \qquad \frac{\Gamma \vdash T_1 \quad \Gamma \vdash T_2 \quad T_1 \parallel T_2}{\Gamma \vdash \langle T_1 \Leftarrow T_2\rangle^{\ell} : T_2 \to T_1}\ \textsc{T\_Cast}$$

$$\frac{\Gamma \vdash e_1 : x{:}T_1 \to T_2 \quad \Gamma \vdash e_2 : T_1}{\Gamma \vdash e_1\, e_2 : T_2\,\{e_2/x\}}\ \textsc{T\_App} \qquad \frac{\Gamma, x{:}T_1 \vdash T_2 \quad \Gamma \vdash e_1 : T_1 \quad \Gamma \vdash e_2 : T_2\,\{e_1/x\}}{\Gamma \vdash (e_1, e_2) : x{:}T_1 \times T_2}\ \textsc{T\_Pair}$$

$$\frac{\Gamma \vdash e : x{:}T_1 \times T_2}{\Gamma \vdash e.1 : T_1}\ \textsc{T\_Proj1} \qquad \frac{\Gamma \vdash e : x{:}T_1 \times T_2}{\Gamma \vdash e.2 : T_2\,\{e.1/x\}}\ \textsc{T\_Proj2}$$

$$\frac{\Gamma \vdash e_1 : \mathsf{Bool} \quad \Gamma \vdash e_2 : T \quad \Gamma \vdash e_3 : T}{\Gamma \vdash \mathsf{if}\, e_1\, \mathsf{then}\, e_2\, \mathsf{else}\, e_3 : T}\ \textsc{T\_If} \qquad \frac{\begin{array}{c}\mathit{TypSpecOf}(C) = x{:}T_1 \rightarrowtail T_2 \rightarrowtail \tau\langle x\rangle \\ \Gamma \vdash e_1 : T_1 \quad \Gamma \vdash e_2 : T_2\,\{e_1/x\} \quad \Gamma \vdash \tau\langle e_1\rangle\end{array}}{\Gamma \vdash C\langle e_1\rangle e_2 : \tau\langle e_1\rangle}\ \textsc{T\_Ctr}$$

$$\frac{\begin{array}{c}\Gamma \vdash e_0 : \tau\langle e\rangle \quad \Gamma \vdash T \quad \mathit{CtrsOf}(\tau) = \overline{C_i}^{\,i \in \{1,\ldots,n\}} \quad \mathit{ArgTypeOf}(\tau) = y{:}T' \\ \text{for all } i,\ \mathit{CtrArgOf}(C_i) = T_i \quad \text{for all } i,\ \Gamma, x_i{:}T_i\,\{e/y\} \vdash e_i : T\end{array}}{\Gamma \vdash \mathsf{match}\, e_0\, \mathsf{with}\, \overline{C_i\, x_i \to e_i}^{\,i \in \{1,\ldots,n\}} : T}\ \textsc{T\_Match}$$

$$\frac{\begin{array}{c}\vdash \Gamma \quad \varnothing \vdash \{x{:}T \mid e_1\} \quad \varnothing \vdash v : T \\ \varnothing \vdash e_2 : \mathsf{Bool} \quad e_1\,\{v/x\} \longrightarrow^{*} e_2\end{array}}{\Gamma \vdash \langle\{x{:}T \mid e_1\}, e_2, v\rangle^{\ell} : \{x{:}T \mid e_1\}}\ \textsc{T\_ACheck} \qquad \frac{\vdash \Gamma \quad \varnothing \vdash \{x{:}T \mid e_1\} \quad \varnothing \vdash e_2 : T}{\Gamma \vdash \langle\!\langle\{x{:}T \mid e_1\}, e_2\rangle\!\rangle^{\ell} : \{x{:}T \mid e_1\}}\ \textsc{T\_WCheck}$$

$$\frac{\vdash \Gamma \quad \varnothing \vdash e : T_1 \quad T_1 \equiv T_2 \quad \varnothing \vdash T_2}{\Gamma \vdash e : T_2}\ \textsc{T\_Conv} \qquad \frac{\vdash \Gamma \quad \varnothing \vdash v : \{x{:}T \mid e\}}{\Gamma \vdash v : T}\ \textsc{T\_Forget}$$

$$\frac{\begin{array}{c}\vdash \Gamma \quad \varnothing \vdash \{x{:}T \mid e\} \quad \varnothing \vdash v : T \\ e\,\{v/x\} \longrightarrow^{*} \mathsf{true}\end{array}}{\Gamma \vdash v : \{x{:}T \mid e\}}\ \textsc{T\_Exact}$$

$\boxed{T_1 \parallel T_2}$ **Type Compatibility**

$$\frac{T_1 \parallel T_2}{\{x{:}T_1 \mid e_1\} \parallel T_2}\ (\textsc{C\_RefineL}) \qquad \frac{\begin{array}{c}\mathit{TypDefOf}(\tau_1) = (\mathsf{type}\, \tau_1\, \langle x{:}T\rangle = \overline{C_i \parallel D_i : T_i}^{\,i}) \\ \text{for all } i,\ \mathit{TypNameOf}(D_i) = \tau_2\end{array}}{\tau_1\langle e_1\rangle \parallel \tau_2\langle e_2\rangle}\ (\textsc{C\_Datatype})$$

Figure 2: Type system.

2 Properties of Type/Term Equivalence

Lemma 1 (Type and Term Equivalences are Equivalences)**.**

(1) The relation $\equiv$ over types is a equivalence relation:

- $T \equiv T$ *for any* T.
- *If* $T_1 \equiv T_2$ *and* $T_2 \equiv T_3$, *then* $T_1 \equiv T_3$.
- *If* $T_1 \equiv T_2$, *then* $T_2 \equiv T_1$.

(2) The relation $\equiv$ *over terms is a equivalence relation:*

- $e \equiv e$ *for any* e.
- *If* $e_1 \equiv e_2$ *and* $e_2 \equiv e_3$, *then* $e_1 \equiv e_3$.
- *If* $e_1 \equiv e_2$, *then* $e_2 \equiv e_1$.

Proof. Since $\equiv$ is the transitive and symmetric closure of $\Rrightarrow$, transitivity and symmetry hold obviously.

We show reflexivity of $\equiv$ over types. Let T be a type, and x be a variable such that $x \notin \mathsf{FV}(T)$. Suppose that $e_1 \longrightarrow e_2$ for some e_1 and e_2 (e.g., $e_1 = \lambda x{:}\mathsf{Bool}.x$ and $e_2 = \mathsf{true}$). Then, we have $T\{e_1/x\} \Rrightarrow T\{e_2/x\}$. Since $T\{e_1/x\} = T\{e_2/x\} = T$, we finish.

Reflexivity of $\equiv$ over terms can be shown similarly. Let e be a term, and x be a variable such that $x \notin \mathsf{FV}(e)$. Suppose that $e_1 \longrightarrow e_2$ for some e_1 and e_2 (e.g., $e_1 = \lambda x{:}\mathsf{Bool}.x$ and $e_2 = \mathsf{true}$). Then, we have $e\{e_1/x\} \Rrightarrow e\{e_2/x\}$. Since $e\{e_1/x\} = e\{e_2/x\} = e$, we finish. □

Lemma 2. *If* $e_1 \longrightarrow e_2$, *then* $e_1 \Rrightarrow e_2$.

Proof. Obvious because $x\{e_1/x\} \Rrightarrow x\{e_2/x\}$. □

Lemma 3.

(1) If $e_1 \Rrightarrow e_2$, *then* $T\{e_1/x\} \Rrightarrow T\{e_2/x\}$.

(2) If $e_1 \Rrightarrow^* e_2$, *then* $T\{e_1/x\} \Rrightarrow^* T\{e_2/x\}$.

(3) If $e_1 \equiv e_2$, *then* $T\{e_1/x\} \equiv T\{e_2/x\}$.

Proof.

1. Since $e_1 \Rrightarrow e_2$, there exist e, y, e_1' and e_2' such that $e_1 = e\{e_1'/y\}$ and $e_2 = e\{e_2'/y\}$ and $e_1' \longrightarrow e_2'$. Suppose that z is a fresh variable. Here, we have
   - $T\{e_1/x\} = T\{e\{e_1'/y\}/x\} = T\{e\{z/y\}\{e_1'/z\}/x\} = T\{e\{z/y\}/x\}\{e_1'/z\}$,
   - $T\{e\{z/y\}/x\}\{e_1'/z\} \Rrightarrow T\{e\{z/y\}/x\}\{e_2'/z\}$, and
   - $T\{e\{z/y\}/x\}\{e_2'/z\} = T\{e\{z/y\}\{e_2'/z\}/x\} = T\{e\{e_2'/y\}/x\} = T\{e_2/x\}$.

   Thus, $T\{e_1/x\} \Rrightarrow T\{e_2/x\}$.
2. By mathematical induction on the number of steps of $e_1 \Rrightarrow^* e_2$.

   Case 0: Obvious because $e_1 = e_2$.

   Case $i+1$: We are given $e_1 \Rrightarrow e_3 \Rrightarrow^i e_2$ for some e_3. By the IH and the first case, we finish.
3. By induction on $e_1 \equiv e_2$.

   Case $e_1 \Rrightarrow e_2$: By the first case.

   Case transitivity and symmetry: By the IH(s). □

Lemma 4.

(1) If $T_1 \Rrightarrow T_2$, *then* $T_1\{e/x\} \Rrightarrow T_2\{e/x\}$

(2) If $T_1 \Rrightarrow^* T_2$, *then* $T_1\{e/x\} \Rrightarrow^* T_2\{e/x\}$

(3) If $T_1 \equiv T_2$, *then* $T_1\{e/x\} \equiv T_2\{e/x\}$.

Proof.

1. By definition, there exist T, y, e_1 and e_2 such that $T_1 = T\{e_1/y\}$ and $T_2 = T\{e_2/y\}$ and $e_1 \longrightarrow e_2$. Suppose that z is a fresh variable. Since the evaluation relation is defined over closed terms, it is found that e_1 and e_2 are closed. Here, we have

   - $T_1\{e/x\} = T\{e_1/y\}\{e/x\} = T\{z/y\}\{e_1/z\}\{e/x\} = T\{z/y\}\{e/x\}\{e_1/z\}$,
   - $T\{z/y\}\{e/x\}\{e_1/z\} \Rrightarrow T\{z/y\}\{e/x\}\{e_2/z\}$, and
   - $T\{z/y\}\{e/x\}\{e_2/z\} = T\{z/y\}\{e_2/z\}\{e/x\} = T\{e_2/y\}\{e/x\} = T_2\{e/x\}$.

   Thus, $T_1\{e/x\} \Rrightarrow T_2\{e/x\}$.

2. By mathematical induction on the number of steps of $T_1 \Rrightarrow^* T_2$.

   Case 0: Obvious because $T_1 = T_2$.

   Case $i+1$: We are given $T_1 \Rrightarrow T_3 \Rrightarrow^i T_2$ for some T_3. By the IH and the first case, we finish.

3. By induction on $T_1 \equiv T_2$.

   Case $T_1 \Rrightarrow T_2$: By the first case.

   Case transitivity and symmetry: Obvious by the IH(s). □

Lemma 5.

(1) If $e_1 \Rrightarrow e_2$, then $e\{e_1/x\} \Rrightarrow e\{e_2/x\}$.

(2) If $e_1 \Rrightarrow^* e_2$, then $e\{e_1/x\} \Rrightarrow^* e\{e_2/x\}$.

(3) If $e_1 \equiv e_2$, then $e\{e_1/x\} \equiv e\{e_2/x\}$

Proof.

1. Since $e_1 \Rrightarrow e_2$, there exists some e', y, e'_1 and e'_2 such that $e_1 = e'\{e'_1/y\}$ and $e_2 = e'\{e'_2/y\}$ and $e'_1 \longrightarrow e'_2$. Suppose that z is a fresh variable. Here, we have

   - $e\{e_1/x\} = e\{e'\{e'_1/y\}/x\} = e\{e'\{z/y\}\{e'_1/z\}/x\} = e\{e'\{z/y\}/x\}\{e'_1/z\}$,
   - $e\{e'\{z/y\}/x\}\{e'_1/z\} \Rrightarrow e\{e'\{z/y\}/x\}\{e'_2/z\}$, and
   - $e\{e'\{z/y\}/x\}\{e'_2/z\} = e\{e'\{z/y\}\{e'_2/z\}/x\} = e\{e'\{e'_2/y\}/x\} = e\{e_2/x\}$.

   Thus, $e\{e_1/x\} \Rrightarrow e\{e_2/x\}$.

2. By mathematical induction on the number of steps of $e_1 \Rrightarrow^* e_2$.

   Case 0: Obvious because $e_1 = e_2$.

   Case $i+1$: We are given $e_1 \Rrightarrow e_3 \Rrightarrow^i e_2$ for some e_3. By the IH and the first case, we finish.

3. By induction on $e_1 \equiv e_2$.

   Case $e_1 \Rrightarrow e_2$: By the first case.

   Case transitivity and symmetry: By the IH(s). □

3 Cotermination

Lemma 6 (Determinism)**.** *If $e \longrightarrow e_1$ and $e \longrightarrow e_2$, then $e_1 = e_2$.*

Proof. Straightforward. □

Lemma 7 (Value Construction Closed Substitution)**.** *For any v, x, and e, $v\,\{e/x\}$ is a value.*

Proof. By structural induction on v.

Case $v = c$, $\texttt{fix}\ f(x{:}T) = e$ or $\langle T_1 \Leftarrow T_2 \rangle^{\ell}$: Obvious.

Case $v = (v_1, v_2)$ or $C\langle e' \rangle v'$: By the IHs. □

Lemma 8. *If e_1 is not a value and $e_2\,\{e_1/x\}$ is, then e_2 is a value.*

Proof. By structural induction on e_2.

Case $e_2 = y$: If $x = y$, then $e_2\,\{e_1/x\} = e_1$, which leads to a contradiction from the assumptions that e_1 is not a value and $e_2\,\{e_1/x\}$ is. Otherwise, if $x \neq y$, then there is a contradiction because $e_2\,\{e_1/x\}$ is a value but $e_2\,\{e_1/x\} = y$ is not.

Case $e_2 = v$: By Lemma 7.

Case $e_2 = e'_1\, e'_2$, $e.i$, $\mathsf{match}\, e'_0\, \mathsf{with}\, \overline{C_i\, y_i \to e'_i}^{\,i}$, $\mathsf{if}\, e'_1\, \mathsf{then}\, e'_2\, \mathsf{else}\, e'_3$, $\Uparrow\ell$, $\langle \{y{:}T \mid e'_1\}, e'_2, v' \rangle^{\ell}$, or $\langle\!\langle \{y{:}T \mid e'_1\}, e'_2 \rangle\!\rangle^{\ell}$: Contradictory.

Case $e = (e_1, e_2)$ or $C\langle e_1 \rangle v_2$: By the IH(s).

□

Lemma 9. *Let e_1 and e_2 are closed terms such that $e_1 \equiv e_2$. If $(v_1\, v_2)\,\{e_1/x\} \longrightarrow e$, then $(v_1\, v_2)\,\{e_2/x\} \longrightarrow e'\,\{e_2/x\}$ for some e' such that $e = e'\,\{e_1/x\}$.*

Proof. By Lemma 7, $v_1\,\{e_1/x\}$, $v_1\,\{e_2/x\}$, $v_2\,\{e_1/x\}$ and $v_2\,\{e_2/x\}$ are values. We proceed by case analysis on v_1. Note that v_1 takes the form of either lambda abstraction or cast since $(v_1\, v_2)\,\{e_1/x\}$ takes a step and that if $(v_1\, v_2)\,\{e_1/x\}$ is closed, then so is $(v_1\, v_2)\,\{e_2/x\}$. In the following, let $i \in \{1, 2\}$.

Case $v_1 = \texttt{fix}\ f(y{:}T) = e'$: Without loss of generality, we can suppose that y and f are fresh. By (E_Red)/(R_Beta),

$$((\texttt{fix}\ f(y{:}T) = e')\, v_2)\,\{e_i/x\} \longrightarrow e'\,\{e_i/x\}\,\{v_2\,\{e_i/x\}/y, v_1\,\{e_i/x\}/f\}.$$

Because $e'\,\{e_i/x\}\,\{v_2\,\{e_i/x\}/y, v_1\,\{e_i/x\}/f\} = e'\,\{v_2/y, v_1/f\}\,\{e_i/x\}$, we finish.

Case $v_1 = \langle \mathsf{Bool} \Leftarrow \mathsf{Bool} \rangle^{\ell}$: Obvious because $(\langle \mathsf{Bool} \Leftarrow \mathsf{Bool} \rangle^{\ell}\, v_2)\,\{e_i/x\} \longrightarrow v_2\,\{e_i/x\}$ by (E_Red)/(R_Base).

Case $v_1 = \langle y{:}T_{11} \to T_{12} \Leftarrow y{:}T_{21} \to T_{22} \rangle^{\ell}$: Without loss of generality, we can suppose that y is fresh. By (E_Red)/(R_Fun),

$$\begin{array}{l}(\langle y{:}T_{11} \to T_{12} \Leftarrow y{:}T_{21} \to T_{22} \rangle^{\ell}\, v_2)\,\{e_i/x\} \longrightarrow \\ \quad \lambda y{:}T_{11}\,\{e_i/x\}.(\lambda z{:}T_{21}\,\{e_i/x\}.\langle T_{12}\,\{e_i/x\} \Leftarrow T_{22}\,\{e_i/x\}\,\{z/y\} \rangle^{\ell}\,(v_2\,\{e_i/x\}\, z))\,(\langle T_{21}\,\{e_i/x\} \Leftarrow T_{11}\,\{e_i/x\} \rangle^{\ell}\, y) \\ = (\lambda y{:}T_{11}.(\lambda z{:}T_{21}.\langle T_{12} \Leftarrow T_{22}\,\{z/y\} \rangle^{\ell}\,(v_2\, z))\,(\langle T_{21} \Leftarrow T_{11} \rangle^{\ell}\, y))\,\{e_i/x\}\end{array}$$

for some fresh variable z. Thus, we finish.

Case $v_1 = \langle y{:}T_{11} \times T_{12} \Leftarrow y{:}T_{21} \times T_{22} \rangle^{\ell}$: Without loss of generality, we can suppose that y is fresh. It is found that $v_2 = (v'_1, v'_2)$ for some v'_1 and v'_2 because (1) $(\langle y{:}T_{11} \times T_{12} \Leftarrow y{:}T_{21} \times T_{22} \rangle^{\ell}\, v_2)\,\{e_1/x\}$ takes a step, (2) the only rule applicable to the application term is (E_Red)/(R_Prod), and (3) v_2 is a value (thus not a variable). By (E_Red)/(R_Prod),

$$\begin{array}{l}(\langle y{:}T_{11} \times T_{12} \Leftarrow y{:}T_{21} \times T_{22} \rangle^{\ell}\,(v'_1, v'_2))\,\{e_i/x\} \longrightarrow \\ \quad (\lambda y{:}T_{11}\,\{e_i/x\}.(y, \langle T_{12}\,\{e_i/x\} \Leftarrow T_{22}\,\{e_i/x\}\,\{v'_1\,\{e_i/x\}/y\} \rangle^{\ell}\, v'_2\,\{e_i/x\}))\,(\langle T_{11}\,\{e_i/x\} \Leftarrow T_{21}\,\{e_i/x\} \rangle^{\ell}\, v'_1\,\{e_i/x\}) \\ = ((\lambda y{:}T_{11}.(y, \langle T_{12} \Leftarrow T_{22}\,\{v'_1/y\} \rangle^{\ell}\, v'_2))\,(\langle T_{11} \Leftarrow T_{21} \rangle^{\ell}\, v'_1))\,\{e_i/x\}.\end{array}$$

Case $v_1 = \langle T_1 \Leftarrow \{y{:}T_2 \,|\, e\}\rangle^\ell$: By (E_Red)/(R_Forget),

$$(\langle T_1 \Leftarrow \{y{:}T_2 \,|\, e\}\rangle^\ell\, v_2)\,\{e_i/x\} \longrightarrow \langle T_1\,\{e_i/x\} \Leftarrow T_2\,\{e_i/x\}\rangle^\ell\, v_2\,\{e_i/x\} = (\langle T_1 \Leftarrow T_2\rangle^\ell\, v_2)\,\{e_i/x\}.$$

Case $v_1 = \langle \{y{:}T_1 \,|\, e\} \Leftarrow T_2\rangle^\ell$ where T_2 is not a refinement type: By (E_Red)/(R_PreCheck),

$$\begin{array}{rcl}(\langle \{y{:}T_1 \,|\, e\} \Leftarrow T_2\rangle^\ell\, v_2)\,\{e_i/x\} & \longrightarrow & \langle\!\langle \{y{:}T_1 \,|\, e\}\,\{e_i/x\}, \langle T_1\,\{e_i/x\} \Leftarrow T_2\,\{e_i/x\}\rangle^\ell\, v_2\,\{e_i/x\}\rangle\!\rangle^\ell \\ & = & \langle\!\langle \{y{:}T_1 \,|\, e\}, \langle T_1 \Leftarrow T_2\rangle^\ell\, v_2\rangle\!\rangle^\ell\,\{e_i/x\}.\end{array}$$

Case $v_1 = \langle \tau_1\langle e_1''\rangle \Leftarrow \tau_2\langle e_2''\rangle\rangle^\ell$: There are three reduction rules by which $(v_1\, v_2)\,\{e_1/x\}$ takes a step.

Case (E_Red)/(R_Datatype): We find that $v_2 = C_2\langle e''\rangle v''$ for some C_2, e'' and v'' since v_2 is a value (thus not a variable). We are given

$$\begin{array}{l}(\langle \tau_1\langle e_1''\rangle \Leftarrow \tau_2\langle e_2''\rangle\rangle^\ell\, C_2\langle e''\rangle v'')\,\{e_1/x\} \longrightarrow \\ \quad C_1\langle e_1''\,\{e_1/x\}\rangle(\langle T_1'\,\{e_1''\,\{e_1/x\}/y_1\} \Leftarrow T_2'\,\{e_2''\,\{e_1/x\}/y_2\}\rangle^\ell\, v''\,\{e_1/x\}) \\ \quad = (C_1\langle e_1''\rangle(\langle T_1'\,\{e_1''/y_1\} \Leftarrow T_2'\,\{e_2''/y_2\}\rangle^\ell\, v''))\,\{e_1/x\}\end{array}$$

where $\delta((\langle \tau_1\langle e_1''\rangle \Leftarrow \tau_2\langle e_2''\rangle\rangle^\ell\, C_2\langle e''\rangle v'')\,\{e_1/x\}) = C_1$ and, for $j \in \{1,2\}$, $\mathit{ArgTypeOf}(\tau_j) = y_j{:}T_j$ and $\mathit{CtrArgOf}(C_j) = T_j'$. Note that only y_1 and y_2 can occur free in T_1' and T_2', respectively, because of well-formedness of the type definition environment. Since $e_1 \equiv e_2$, we have $(v_1\, v_2)\,\{e_1/x\} \equiv (v_1\, v_2)\,\{e_2/x\}$ by Lemma 5 (3). From well-formedness of the constructor choice function, we have $\delta((v_1\, v_2)\,\{e_2/x\}) = \delta((v_1\, v_2)\,\{e_1/x\}) = C_1$. Thus, by (E_Red)/(R_Datatype),

$$\begin{array}{l}(\langle \tau_1\langle e_1''\rangle \Leftarrow \tau_2\langle e_2''\rangle\rangle^\ell\, C_2\langle e''\rangle v'')\,\{e_2/x\} \longrightarrow \\ \quad C_1\langle e_1''\,\{e_2/x\}\rangle(\langle T_1'\,\{e_1''\,\{e_2/x\}/y_1\} \Leftarrow T_2'\,\{e_2''\,\{e_2/x\}/y_2\}\rangle^\ell\, v''\,\{e_2/x\}) \\ \quad = (C_1\langle e_1''\rangle(\langle T_1'\,\{e_1''/y_1\} \Leftarrow T_2'\,\{e_2''/y_2\}\rangle^\ell\, v''))\,\{e_2/x\}.\end{array}$$

Case (E_Red)/(R_DatatypeMono): By (E_Red)/(R_DatatypeMono), $(\langle \tau_1 \Leftarrow \tau_2\rangle^\ell\, v_2)\,\{e_i/x\} \longrightarrow v_2\,\{e_i/x\}$.

Case (E_Red)/(R_DatatypeFail): We are given $(\langle \tau_1\langle e_1''\rangle \Leftarrow \tau_2\langle e_2''\rangle\rangle^\ell\, v_2)\,\{e_1/x\} \longrightarrow \Uparrow\ell$ and $\delta((\langle \tau_1\langle e_1''\rangle \Leftarrow \tau_2\langle e_2''\rangle\rangle^\ell\, v_2)\,\{e_1/x\})$ is undefined. Since $e_1 \equiv e_2$, we have $(v_1\, v_2)\,\{e_1/x\} \equiv (v_1\, v_2)\,\{e_2/x\}$ by Lemma 5 (3). If $\delta((v_1\, v_2)\,\{e_2/x\})$ is defined, then so is $\delta((v_1\, v_2)\,\{e_1/x\})$ from well-formedness of the constructor choice function but it contradicts. Thus, $\delta((v_1\, v_2)\,\{e_2/x\})$ is also undefined and so, by (E_Red)/(R_DatatypeFail), $(\langle \tau_1\langle e_1''\rangle \Leftarrow \tau_2\langle e_2''\rangle\rangle^\ell\, v_2)\,\{e_2/x\} \longrightarrow \Uparrow\ell$. ☐

Lemma 10. *Let e_1 and e_2 be terms such that $e_1 \longrightarrow e_2$.*

(1) If $(v_1\, v_2)\,\{e_1/x\} \longrightarrow e$, then $(v_1\, v_2)\,\{e_2/x\} \longrightarrow e'\,\{e_2/x\}$ for some e' such that $e = e'\,\{e_1/x\}$.

(2) If $(v_1\, v_2)\,\{e_2/x\} \longrightarrow e$, then $(v_1\, v_2)\,\{e_1/x\} \longrightarrow e'\,\{e_1/x\}$ for some e' such that $e = e'\,\{e_2/x\}$.

Proof. Since the evaluation relation is defined over closed terms, e_1 and e_2 are closed. Thus, we finish by Lemma 9. ☐

Lemma 11. *Let e_1 and e_2 are closed terms, and $i \in \{1,2\}$. If $(v.i)\,\{e_1/x\} \longrightarrow e$, then $(v.i)\,\{e_2/x\} \longrightarrow e'\,\{e_2/x\}$ for some e' such that $e = e'\,\{e_1/x\}$.*

Proof. By Lemma 7, $v\,\{e_1/x\}$ and $v\,\{e_2/x\}$ are values. We find that v takes the form of pair since $(v.i)\,\{e_1/x\}$ takes a step. Note that if $(v.i)\,\{e_1/x\}$ is closed, then so is $(v.i)\,\{e_2/x\}$.

We are given $v = (v_1, v_2)$ for some v_1 and v_2. By (E_Red)/(R_Proj$i$), for $j \in \{1,2\}$,

$$((v_1, v_2).i)\,\{e_j/x\} \longrightarrow v_i\,\{e_j/x\}.$$

Thus, we finish. ☐

Lemma 12. *Let e_1 and e_2 be terms such that $e_1 \longrightarrow e_2$, and $i \in \{1,2\}$.*

(1) If $(v.i)\,\{e_1/x\} \longrightarrow e$, then $(v.i)\,\{e_2/x\} \longrightarrow e'\,\{e_2/x\}$ for some e' such that $e = e'\,\{e_1/x\}$.

(2) *If* $(v.i)\,\{e_2/x\} \longrightarrow e$, *then* $(v.i)\,\{e_1/x\} \longrightarrow e'\,\{e_1/x\}$ *for some* e' *such that* $e = e'\,\{e_2/x\}$.

Proof. Since the evaluation relation is defined over closed terms, e_1 and e_2 are closed. Thus, we finish by Lemma 11. $\square$

Lemma 13. *Let* e_1 *and* e_2 *are closed terms. If* $(\mathsf{if}\, v\, \mathsf{then}\, e'_1\, \mathsf{else}\, e'_2)\,\{e_1/x\} \longrightarrow e$, *then* $(\mathsf{if}\, v\, \mathsf{then}\, e'_1\, \mathsf{else}\, e'_2)\,\{e_2/x\} \longrightarrow e'\,\{e_2/x\}$ *for some* e' *such that* $e = e'\,\{e_1/x\}$.

Proof. By Lemma 7, $v\,\{e_1/x\}$ and $v\,\{e_2/x\}$ are values. Note that v takes the form of Boolean value since $(\mathsf{if}\, v\, \mathsf{then}\, e'_1\, \mathsf{else}\, e'_2)\,\{e_1/x\}$ takes a step and that if $(\mathsf{if}\, v\, \mathsf{then}\, e'_1\, \mathsf{else}\, e'_2)\,\{e_1/x\}$ is closed, then so is $(\mathsf{if}\, v\, \mathsf{then}\, e'_1\, \mathsf{else}\, e'_2)\,\{e_2/x\}$. By case analysis on v. In the following, let $i \in \{1, 2\}$.

Case $v = \mathsf{true}$: By (E_Red)/(R_IfTrue),

$$(\mathsf{if}\,\mathsf{true}\,\mathsf{then}\, e'_1\, \mathsf{else}\, e'_2)\,\{e_i/x\} \longrightarrow e'_1\,\{e_i/x\}.$$

Case $v = \mathsf{false}$: By (E_Red)/(R_IfFalse),

$$(\mathsf{if}\,\mathsf{false}\,\mathsf{then}\, e'_1\, \mathsf{else}\, e'_2)\,\{e_i/x\} \longrightarrow e'_2\,\{e_i/x\}.$$

$\square$

Lemma 14. *Let* e_1 *and* e_2 *be terms such that* $e_1 \longrightarrow e_2$.

(1) *If* $(\mathsf{if}\, v\, \mathsf{then}\, e'_1\, \mathsf{else}\, e'_2)\,\{e_1/x\} \longrightarrow e$, *then* $(\mathsf{if}\, v\, \mathsf{then}\, e'_1\, \mathsf{else}\, e'_2)\,\{e_2/x\} \longrightarrow e'\,\{e_2/x\}$ *for some* e' *such that* $e = e'\,\{e_1/x\}$.

(2) *If* $(\mathsf{if}\, v\, \mathsf{then}\, e'_1\, \mathsf{else}\, e'_2)\,\{e_2/x\} \longrightarrow e$, *then* $(\mathsf{if}\, v\, \mathsf{then}\, e'_1\, \mathsf{else}\, e'_2)\,\{e_1/x\} \longrightarrow e'\,\{e_1/x\}$ *for some* e' *such that* $e = e'\,\{e_2/x\}$.

Proof. Since the evaluation relation is defined over closed terms, e_1 and e_2 are closed. Thus, we finish by Lemma 13. $\square$

Lemma 15. *Let* e_1 *and* e_2 *are closed terms. If* $(\mathsf{match}\, v\, \mathsf{with}\, \overline{C_i\, y_i \rightarrow e'_i}^{\,i})\,\{e_1/x\} \longrightarrow e$, *then* $(\mathsf{match}\, v\, \mathsf{with}\, \overline{C_i\, y_i \rightarrow e'_i}^{\,i})\,\{e_2/x\} \longrightarrow e'\,\{e_2/x\}$ *for some* e' *such that* $e = e'\,\{e_1/x\}$.

Proof. Without loss of generality, we can suppose that each y_i is fresh. By Lemma 7, $v\,\{e_1/x\}$ and $v\,\{e_2/x\}$ are values. We find that v takes the form of constructor application since $(\mathsf{match}\, v\, \mathsf{with}\, \overline{C_i\, y_i \rightarrow e'_i}^{\,i})\,\{e_1/x\}$ takes a step. Note that if $(\mathsf{match}\, v\, \mathsf{with}\, \overline{C_i\, y_i \rightarrow e'_i}^{\,i})\,\{e_1/x\}$ is closed, then so is $(\mathsf{match}\, v\, \mathsf{with}\, \overline{C_i\, y_i \rightarrow e'_i}^{\,i})\,\{e_2/x\}$.

We are given $v = C_j\langle e'\rangle v'$ for some $C_j \in \overline{C_i}^{\,i}$, e' and v'. By (E_Red)/(R_Match), for $k \in \{1, 2\}$,

$$\begin{array}{rcl}(\mathsf{match}\, C_j\langle e'\rangle v'\, \mathsf{with}\, \overline{C_i\, y_i \rightarrow e'_i}^{\,i})\,\{e_k/x\} & \longrightarrow & e'_j\,\{e_k/x\}\,\{v'\,\{e_k/x\}/y_j\} \\ & = & e'_j\,\{v'/y_j\}\,\{e_k/x\}.\end{array}$$

Thus, we finish. $\square$

Lemma 16. *Let* e_1 *and* e_2 *be terms such that* $e_1 \longrightarrow e_2$.

(1) *If* $(\mathsf{match}\, v\, \mathsf{with}\, \overline{C_i\, y_i \rightarrow e'_i}^{\,i})\,\{e_1/x\} \longrightarrow e$, *then* $(\mathsf{match}\, v\, \mathsf{with}\, \overline{C_i\, y_i \rightarrow e'_i}^{\,i})\,\{e_2/x\} \longrightarrow e'\,\{e_2/x\}$ *for some* e' *such that* $e = e'\,\{e_1/x\}$.

(2) *If* $(\mathsf{match}\, v\, \mathsf{with}\, \overline{C_i\, y_i \rightarrow e'_i}^{\,i})\,\{e_2/x\} \longrightarrow e$, *then* $(\mathsf{match}\, v\, \mathsf{with}\, \overline{C_i\, y_i \rightarrow e'_i}^{\,i})\,\{e_1/x\} \longrightarrow e'\,\{e_1/x\}$ *for some* e' *such that* $e = e'\,\{e_2/x\}$.

Proof. Since the evaluation relation is defined over closed terms, e_1 and e_2 are closed. Thus, we finish by Lemma 15. $\square$

Lemma 17. *Let* e_1 *and* e_2 *are closed terms. If* $\langle\!\langle \{y{:}T \mid e'_1\}, v\rangle\!\rangle^{\ell}\,\{e_1/x\} \longrightarrow e$, *then* $\langle\!\langle \{y{:}T \mid e'_1\}, v\rangle\!\rangle^{\ell}\,\{e_2/x\} \longrightarrow e'\,\{e_2/x\}$ *for some* e' *such that* $e = e'\,\{e_1/x\}$.

Proof. Without loss of generality, we can suppose that y is fresh. By Lemma 7, $v\,\{e_1/x\}$ and $v\,\{e_2/x\}$ are values. Note that if $\langle\!\langle \{y{:}T\,|\,e'_1\}, v\rangle\!\rangle^\ell\,\{e_1/x\}$ is closed, then so is $\langle\!\langle \{y{:}T\,|\,e'_1\}, v\rangle\!\rangle^\ell\,\{e_2/x\}$. Letting $i \in \{1,2\}$, by (E_Red)/(R_Check),

$$\begin{array}{rcl}\langle\!\langle \{y{:}T\,|\,e'_1\}, v\rangle\!\rangle^\ell\,\{e_i/x\} & \longrightarrow & \langle \{y{:}T\,|\,e'_1\}\,\{e_i/x\}, e'_1\,\{e_i/x\}\,\{v\,\{e_i/x\}/y\}, v\,\{e_i/x\}\rangle^\ell \\ & = & \langle \{y{:}T\,|\,e'_1\}, e'_1\,\{v/y\}, v\rangle^\ell\,\{e_i/x\}.\end{array}$$

Thus, we finish. □

Lemma 18. *Let e_1 and e_2 be terms such that $e_1 \longrightarrow e_2$.*

(1) If $\langle\!\langle \{y{:}T\,|\,e'_1\}, v\rangle\!\rangle^\ell\,\{e_1/x\} \longrightarrow e$, then $\langle\!\langle \{y{:}T\,|\,e'_1\}, v\rangle\!\rangle^\ell\,\{e_2/x\} \longrightarrow e'\,\{e_2/x\}$ for some e' such that $e = e'\,\{e_1/x\}$.

(2) If $\langle\!\langle \{y{:}T\,|\,e'_1\}, v\rangle\!\rangle^\ell\,\{e_2/x\} \longrightarrow e$, then $\langle\!\langle \{y{:}T\,|\,e'_1\}, v\rangle\!\rangle^\ell\,\{e_1/x\} \longrightarrow e'\,\{e_1/x\}$ for some e' such that $e = e'\,\{e_2/x\}$.

Proof. Since the evaluation relation is defined over closed terms, e_1 and e_2 are closed. Thus, we finish by Lemma 17. □

Lemma 19. *Let e_1 and e_2 are closed terms. If $\langle \{y{:}T\,|\,e'_1\}, v_1, v_2\rangle^\ell\,\{e_1/x\} \longrightarrow e$, then $\langle \{y{:}T\,|\,e'_1\}, v_1, v_2\rangle^\ell\,\{e_2/x\} \longrightarrow e'\,\{e_2/x\}$ for some e' such that $e = e'\,\{e_1/x\}$.*

Proof. By Lemma 7, $v_1\,\{e_1/x\}$ and $v_1\,\{e_2/x\}$ are values. Note that v_1 takes the form of Boolean value since $\langle \{y{:}T\,|\,e'_1\}, v_1, v_2\rangle^\ell\,\{e_1/x\}$ takes a step and that if $\langle \{y{:}T\,|\,e'_1\}, v_1, v_2\rangle^\ell\,\{e_1/x\}$ is closed, then so is $\langle \{y{:}T\,|\,e'_1\}, v_1, v_2\rangle^\ell\,\{e_2/x\}$. By case analysis on v_1. In the following, let $i \in \{1,2\}$.

Case $v_1 = \mathsf{true}$: By (E_Red)/(R_OK), $\langle \{y{:}T\,|\,e'_1\}, \mathsf{true}, v_2\rangle^\ell\,\{e_i/x\} \longrightarrow v_2\,\{e_i/x\}$.

Case $v_2 = \mathsf{false}$: By (E_Red)/(R_Fail), $\langle \{y{:}T\,|\,e'_1\}, \mathsf{false}, v_2\rangle^\ell\,\{e_i/x\} \longrightarrow \Uparrow\ell$. □

Lemma 20. *Let e_1 and e_2 be terms such that $e_1 \longrightarrow e_2$.*

(1) If $\langle \{y{:}T\,|\,e'_1\}, v_1, v_2\rangle^\ell\,\{e_1/x\} \longrightarrow e$, then $\langle \{y{:}T\,|\,e'_1\}, v_1, v_2\rangle^\ell\,\{e_2/x\} \longrightarrow e'\,\{e_2/x\}$ for some e' such that $e = e'\,\{e_1/x\}$.

(2) If $\langle \{y{:}T\,|\,e'_1\}, v_1, v_2\rangle^\ell\,\{e_2/x\} \longrightarrow e$, then $\langle \{y{:}T\,|\,e'_1\}, v_1, v_2\rangle^\ell\,\{e_1/x\} \longrightarrow e'\,\{e_1/x\}$ for some e' such that $e = e'\,\{e_2/x\}$.

Proof. Since the evaluation relation is defined over closed terms, e_1 and e_2 are closed. Thus, we finish by Lemma 19. □

Lemma 21.

(1) If $e_1 \longrightarrow^n e_2$ is derived by (E_Red), then $E[e_1] \longrightarrow^n E[e_2]$ is derived by applying only (E_Red).

(2) If $e \longrightarrow^* \Uparrow\ell$, then $E[e] \longrightarrow^* \Uparrow\ell$.

Proof.

1. By induction on the number of evaluation steps of $e_1 \longrightarrow^n e_2$.

   Case 0: Obvious.

   Case $i+1$: We are given $e_1 \longrightarrow e_3 \longrightarrow^i e_2$ for some e_3. Since $e_1 \longrightarrow e_3$ is derived by (E_Red), there exist some E', e'_1 and e'_3 such that $e'_1 \rightsquigarrow e'_3$. Since $E[E'[e'_1]] \longrightarrow E[E'[e'_3]]$ by (E_Red), we finish by the IH.

2. By induction on the number of evaluation steps of $e_1 \longrightarrow^* \Uparrow\ell$.

   Case 0: Since $e = \Uparrow\ell$, we finish by (E_Blame) if $E \neq [\,]$.

   Case $n+1$: We are given $e \longrightarrow e' \longrightarrow^n \Uparrow\ell$ for some e'. If the evaluation rule applied to e is (E_Red), then $e = E'[e_1]$ and $e' = E'[e_2]$ for some E', e_1 and e_2 such that $e_1 \rightsquigarrow e_2$. Since $E[E'[e_1]] \longrightarrow E[E'[e_2]]$ by (E_Red), we finish by the IH. Otherwise, if the evaluation rule applied to e is (E_Blame), then $e = E'[\Uparrow\ell]$ for some E', and $e' = \Uparrow\ell$. By (E_Blame), $E[E'[\Uparrow\ell]] \longrightarrow \Uparrow\ell$.

$\square$

Lemma 22. *Suppose that* $e_1 \longrightarrow e_2$. *If* $e\{e_1/x\} = E_1[\Uparrow\ell]$, *then there exists some* E_2 *such that* $e\{e_2/x\} = E_2[\Uparrow\ell]$.

Proof. By structural induction on e

Case $e = x$: It is found that $e_1 = e\{e_1/x\} = E_1[\Uparrow\ell]$. Since $E_1[\Uparrow\ell] \longrightarrow \Uparrow\ell$ by (E_Blame), $e_2 = \Uparrow\ell$.

Case $e = v$: Contradictory.

Case $e = \Uparrow\ell'$: If $\ell' = \ell$, then obvious. Otherwise, if $\ell' \neq \ell$, then contradictory since $e\{e_1/x\} = E_1[\Uparrow\ell]$.

Case $e = e'_1\, e'_2$: Since $e\{e_1/x\} = E_1[\Uparrow\ell]$, there are two cases we have to consider.

Case $E_1 = E'_1\, e'_2\{e_1/x\}$: Since $e'_1\{e_1/x\} = E'_1[\Uparrow\ell]$, there exists some E'_2 such that $e'_1\{e_2/x\} = E'_2[\Uparrow\ell]$, by the IH. Since $E'_2\, e'_2\{e_2/x\}$ is an evaluation context and $e\{e_2/x\} = E'_2[\Uparrow\ell]\, e'_2\{e_2/x\}$, we finish.

Case $E_1 = e'_1\{e_1/x\}\, E'_1$ where $e'_1\{e_1/x\}$ is a value: Since $e'_2\{e_1/x\} = E'_1[\Uparrow\ell]$, there exists some E'_2 such that $e'_2\{e_2/x\} = E'_2[\Uparrow\ell]$, by the IH. Since $e'_1\{e_1/x\}$ is a value and e_1 is not a value from $e_1 \longrightarrow e_2$, it is found by Lemmas 8 and 7 that $e'_1\{e_2/x\}$ is a value. Thus, since $e'_1\{e_2/x\}\, E'_2$ is an evaluation context and $e\{e_2/x\} = e'_1\{e_2/x\}\, E'_2[\Uparrow\ell]$, we finish.

Case $e = (e'_1, e'_2)$ which is a not value: Since $e\{e_1/x\} = E_1[\Uparrow\ell]$, there are two cases we have to consider.

Case $E_1 = (E'_1, e'_2\{e_1/x\})$: Since $e'_1\{e_1/x\} = E'_1[\Uparrow\ell]$, there exists some E'_2 such that $e'_1\{e_2/x\} = E'_2[\Uparrow\ell]$, by the IH. Since $(E'_2, e'_2\{e_2/x\})$ is an evaluation context and $e\{e_2/x\} = (E'_2[\Uparrow\ell], e'_2\{e_2/x\})$, we finish.

Case $E_1 = (e'_1\{e_1/x\}, E'_1)$ where $e'_1\{e_1/x\}$ is a value: Since $e'_2\{e_1/x\} = E'_1[\Uparrow\ell]$, there exists some E'_2 such that $e'_2\{e_2/x\} = E'_2[\Uparrow\ell]$, by the IH. Since $e'_1\{e_1/x\}$ is a value, it is found by Lemmas 8 and 7 that $e'_1\{e_2/x\}$ is a value. Thus, since $(e'_1\{e_2/x\}, E'_2)$ is an evaluation context and $e\{e_2/x\} = e'_1\{e_2/x\}\, E'_2[\Uparrow\ell]$, we finish.

Case $e = e'.i$ ($i \in \{1,2\}$): Since $e\{e_1/x\} = E_1[\Uparrow\ell]$, there exists some E'_1 such that $e'\{e_1/x\} = E'_1[\Uparrow\ell]$. By the IH, there exists some E'_2 such that $e'\{e_2/x\} = E'_2[\Uparrow\ell]$. Since $e\{e_2/x\} = E'_2[\Uparrow\ell].i$, we finish.

Case $e = C\langle e'_1\rangle e'_2$ which is a not value: Since $e\{e_1/x\} = E_1[\Uparrow\ell]$, there exists some E'_1 such that $e'_2\{e_1/x\} = E'_1[\Uparrow\ell]$. By the IH, there exists some E'_2 such that $e'_2\{e_2/x\} = E'_2[\Uparrow\ell]$. Since $C\langle e'_1\{e_2/x\}\rangle E'_2$ is an evaluation context and $e\{e_2/x\} = C\langle e'_1\{e_2/x\}\rangle E'_2[\Uparrow\ell]$, we finish.

Case $e = \mathsf{match}\, e'_0\, \mathsf{with}\, \overline{C_i\, y_i \to e'_i}^{\,i}$: Since $e\{e_1/x\} = E_1[\Uparrow\ell]$, there exists some E'_1 such that $e'_0\{e_1/x\} = E'_1[\Uparrow\ell]$. By the IH, there exists some E'_2 such that $e'_0\{e_2/x\} = E'_2[\Uparrow\ell]$. Since $\mathsf{match}\, E'_2\, \mathsf{with}\, \overline{C_i\, y_i \to e'_i}^{\,i}$ is an evaluation context and $e\{e_2/x\} = \mathsf{match}\, E'_2[\Uparrow\ell]\, \mathsf{with}\, \overline{C_i\, y_i \to e'_i}^{\,i}$, we finish.

Case $e = \mathsf{if}\, e'_1\, \mathsf{then}\, e'_2\, \mathsf{else}\, e'_3$: Since $e\{e_1/x\} = E_1[\Uparrow\ell]$, there exists some E'_1 such that $e'_1\{e_1/x\} = E'_1[\Uparrow\ell]$. By the IH, there exists some E'_2 such that $e'_1\{e_2/x\} = E'_2[\Uparrow\ell]$. Since $\mathsf{if}\, E'_2\, \mathsf{then}\, e'_2\{e_2/x\}\, \mathsf{else}\, e'_3\{e_2/x\}$ is an evaluation context and $e\{e_2/x\} = \mathsf{if}\, E'_2[\Uparrow\ell]\, \mathsf{then}\, e'_2\{e_2/x\}\, \mathsf{else}\, e'_3\{e_2/x\}$, we finish.

Case $e = \langle\!\langle \{y{:}T \mid e'_1\}, e'_2 \rangle\!\rangle^{\ell}$: Since $e\{e_1/x\} = E_1[\Uparrow\ell]$, there exists some E'_1 such that $e'_2\{e_1/x\} = E'_1[\Uparrow\ell]$. By the IH, there exists some E'_2 such that $e'_2\{e_2/x\} = E'_2[\Uparrow\ell]$. Since $\langle\!\langle \{y{:}T \mid e'_1\}\{e_2/x\}, E'_2 \rangle\!\rangle^{\ell}$ is an evaluation context and $e\{e_2/x\} = \langle\!\langle \{y{:}T \mid e'_1\}\{e_2/x\}, E'_2[\Uparrow\ell] \rangle\!\rangle^{\ell}$, we finish.

Case $e = \langle \{y{:}T \mid e'_1\}, e'_2, v' \rangle^{\ell}$: Since $e\{e_1/x\} = E_1[\Uparrow\ell]$, there exists some E'_1 such that $e'_2\{e_1/x\} = E'_1[\Uparrow\ell]$. By the IH, there exists some E'_2 such that $e'_2\{e_2/x\} = E'_2[\Uparrow\ell]$. Since $\langle \{y{:}T \mid e'_1\}\{e_2/x\}, E'_2, v'\{e_2/x\} \rangle^{\ell}$ is an evaluation context by Lemma 7 and $e\{e_2/x\} = \langle \{y{:}T \mid e'_1\}\{e_2/x\}, E'_2[\Uparrow\ell], v'\{e_2/x\} \rangle^{\ell}$, we finish. $\square$

Lemma 23 (Cotermination: Reduction on the Left)**.** *Let* e_1 *and* e_2 *be terms such that* $e_1 \longrightarrow e_2$. *If* $e\{e_1/x\} \longrightarrow e'$, *then* $e\{e_2/x\} \longrightarrow^* e''\{e_2/x\}$ *for some* e'' *such that* $e' = e''\{e_1/x\}$. *Moreover, if* $e\{e_1/x\} \longrightarrow e'$ *is derived by (*E_Red*), then the evaluation* $e\{e_2/x\} \longrightarrow^* e''\{e_2/x\}$ *is derived by applying only (*E_Red*).*

Proof. By structural induction on e. If $e\{e_1/x\} \longrightarrow e'$ is derived by (E_Blame), then there exist some E_1 and ℓ such that $e\{e_1/x\} = E_1[\Uparrow\ell]$ and $e' = \Uparrow\ell$. By Lemma 22, there exists some E_2 such that $e\{e_2/x\} = E_2[\Uparrow\ell]$. Thus, by (E_Blame), $e\{e_2/x\} \longrightarrow \Uparrow\ell$.

In what follows, we suppose that $e\{e_1/x\} \longrightarrow e'$ is derived by (E_Red). We proceed by case analysis on e. Note that e_1 is not a value from $e_1 \longrightarrow e_2$.

Case $e = y$: If $x = y$, then we have $e\{e_1/x\} = e_1$ and $e\{e_2/x\} = e_2$. We finish by letting $e'' = e_2$ because $e\{e_1/x\} = e_1 \longrightarrow e_2$ and $e_2\{e_1/x\} = e_2\{e_2/x\} = e_2$. Note that e_2 is closed since the evaluation relation is defined over closed terms.

Otherwise, if $x \neq y$, then there is a contradiction because the assumption says that $e\{e_1/x\} = y$ takes a step.

Case $e = \Uparrow\ell$: Contradictory.

Case $e = v$: Contradictory by Lemma 7 since $e\{e_1/x\}$ takes a step.

Case $e = e'_1\, e'_2$: Since $e\{e_1/x\}$ takes a step, there are three cases we have to consider.

Case $e'_1\{e_1/x\} \longrightarrow e''$ by (E_Red): By the IH, there exists some e''_1 such that $e'_1\{e_2/x\} \longrightarrow^* e''_1\{e_2/x\}$ and $e'' = e''_1\{e_1/x\}$. Moreover, the evaluation $e'_1\{e_2/x\} \longrightarrow^* e''_1\{e_2/x\}$ is derived by applying only (E_Red). Thus, by Lemma 21 (1), $(e'_1\, e'_2)\{e_1/x\} \longrightarrow (e''_1\, e'_2)\{e_1/x\}$ and $(e'_1\, e'_2)\{e_2/x\} \longrightarrow^* (e''_1\, e'_2)\{e_2/x\}$.

Case $e'_1\{e_1/x\}$ is a value and $e'_2\{e_1/x\} \longrightarrow e''$ by (E_Red): By Lemmas 8 and 7, $e'_1\{e_2/x\}$ is a value. By the IH, there exists some e''_2 such that $e'_2\{e_2/x\} \longrightarrow^* e''_2\{e_2/x\}$ and $e'' = e''_2\{e_1/x\}$. Moreover, the evaluation $e'_2\{e_2/x\} \longrightarrow^* e''_2\{e_2/x\}$ is derived by applying only (E_Red). Thus, by Lemma 21 (1), $(e'_1\, e'_2)\{e_1/x\} \longrightarrow (e'_1\, e''_2)\{e_1/x\}$ and $(e'_1\, e'_2)\{e_2/x\} \longrightarrow^* (e'_1\, e''_2)\{e_2/x\}$.

Case $e'_1\{e_1/x\}$ and $e'_2\{e_1/x\}$ are values: Since e'_1 and e'_2 are values by Lemma 8, we finish by Lemma 10 (1).

Case $e = (e'_1, e'_2)$: Similarly to the case for application term. Since $e\{e_1/x\}$ takes a step, there are two cases we have to consider.

Case $e'_1\{e_1/x\} \longrightarrow e''$ by (E_Red): By the IH, there exists some e''_1 such that $e'_1\{e_2/x\} \longrightarrow^* e''_1\{e_2/x\}$ and $e'' = e''_1\{e_1/x\}$. Moreover, the evaluation $e'_1\{e_2/x\} \longrightarrow^* e''_1\{e_2/x\}$ is derived by applying only (E_Red). Thus, by Lemma 21 (1), $(e'_1, e'_2)\{e_1/x\} \longrightarrow (e''_1, e'_2)\{e_1/x\}$ and $(e'_1, e'_2)\{e_2/x\} \longrightarrow^* (e''_1, e'_2)\{e_2/x\}$.

Case $e'_1\{e_1/x\}$ is a value and $e'_2\{e_1/x\} \longrightarrow e''$ by (E_Red): By Lemmas 8 and 7, $e'_1\{e_2/x\}$ is a value. By the IH, there exists some e''_2 such that $e'_2\{e_2/x\} \longrightarrow^* e''_2\{e_2/x\}$ and $e'' = e''_2\{e_1/x\}$. Moreover, the evaluation $e'_2\{e_2/x\} \longrightarrow^* e''_2\{e_2/x\}$ is derived by applying only (E_Red). Thus, by Lemma 21 (1), $(e'_1, e'_2)\{e_1/x\} \longrightarrow (e'_1, e''_2)\{e_1/x\}$ and $(e'_1, e'_2)\{e_2/x\} \longrightarrow^* (e'_1, e''_2)\{e_2/x\}$.

Case $e = e'.i$ for $i \in \{1,2\}$: Similarly to the case for application term except for use of Lemma 12 (1). Since $e\{e_1/x\}$ takes a step, there are two cases we have to consider.

Case $e'\{e_1/x\} \longrightarrow e''$ by (E_Red): By the IH, there exists some e''' such that $e'\{e_2/x\} \longrightarrow^* e'''\{e_2/x\}$ and $e'' = e'''\{e_1/x\}$. Moreover, the evaluation $e'\{e_2/x\} \longrightarrow^* e'''\{e_2/x\}$ is derived by applying only (E_Red). Thus, by Lemma 21 (1), $(e'.i)\{e_1/x\} \longrightarrow (e'''.i)\{e_1/x\}$ and $(e'.i)\{e_2/x\} \longrightarrow^* (e'''.i)\{e_2/x\}$.

Case $e'\{e_1/x\}$ is a value: Since e' is a value by Lemma 8, we finish by Lemma 12 (1).

Case $e = C\langle e'_1\rangle e'_2$: Similarly to the case for application term. Since $e\{e_1/x\}$ takes a step, it is found that $e'_2\{e_1/x\} \longrightarrow e''$ by (E_Red) for some e''. By the IH, there exists some e''_2 such that $e'_2\{e_2/x\} \longrightarrow^* e''_2\{e_2/x\}$ and $e'' = e''_2\{e_1/x\}$. Moreover, the evaluation $e'_2\{e_2/x\} \longrightarrow^* e''_2\{e_2/x\}$ is derived by applying only (E_Red). Thus, by Lemma 21 (1), $(C\langle e'_1\rangle e'_2)\{e_1/x\} \longrightarrow (C\langle e'_1\rangle e''_2)\{e_1/x\}$ and $(C\langle e'_1\rangle e'_2)\{e_2/x\} \longrightarrow^* (C\langle e'_1\rangle e''_2)\{e_2/x\}$.

Case $e = \mathsf{match}\, e'_0\, \mathsf{with}\, \overline{C_i\, y_i \rightarrow e'_i}^{\,i}$: Similarly to the case for application term except for use of Lemma 16 (1). Since $e\{e_1/x\}$ takes a step, there are two cases we have to consider.

Case $e'_0\,\{e_1/x\} \longrightarrow e''$ by (E_Red): By the IH, there exists some e''_0 such that $e'_0\,\{e_2/x\} \longrightarrow^* e''_0\,\{e_2/x\}$ and $e'' = e''_0\,\{e_1/x\}$. Moreover, the evaluation $e'_0\,\{e_2/x\} \longrightarrow^* e''_0\,\{e_2/x\}$ is derived by applying only (E_Red). Thus, by Lemma 21 (1),

$$\begin{array}{rcl}(\mathsf{match}\,e'_0\,\mathsf{with}\,\overline{C_i\,y_i \rightarrow e'_i}^{\,i})\,\{e_1/x\} & \longrightarrow & (\mathsf{match}\,e''_0\,\mathsf{with}\,\overline{C_i\,y_i \rightarrow e'_i}^{\,i})\,\{e_1/x\} \\ (\mathsf{match}\,e'_0\,\mathsf{with}\,\overline{C_i\,y_i \rightarrow e'_i}^{\,i})\,\{e_2/x\} & \longrightarrow^* & (\mathsf{match}\,e''_0\,\mathsf{with}\,\overline{C_i\,y_i \rightarrow e'_i}^{\,i})\,\{e_2/x\}.\end{array}$$

Case $e'_0\,\{e_1/x\}$ is a value: Since e'_0 is a value by Lemma 8, we finish by Lemma 16 (1).

Case $e = \mathsf{if}\,e'_1\,\mathsf{then}\,e'_2\,\mathsf{else}\,e'_3$: Similarly to the case for application term except for use of Lemma 14 (1). Since $e\,\{e_1/x\}$ takes a step, there are two cases we have to consider.

Case $e'_1\,\{e_1/x\} \longrightarrow e''$ by (E_Red): By the IH, there exists some e''_1 such that $e'_1\,\{e_2/x\} \longrightarrow^* e''_1\,\{e_2/x\}$ and $e'' = e''_1\,\{e_1/x\}$. Moreover, the evaluation $e'_1\,\{e_2/x\} \longrightarrow^* e''_1\,\{e_2/x\}$ is derived by applying only (E_Red). Thus, by Lemma 21 (1),

$$\begin{array}{rcl}(\mathsf{if}\,e'_1\,\mathsf{then}\,e'_2\,\mathsf{else}\,e'_3)\,\{e_1/x\} & \longrightarrow & (\mathsf{if}\,e''_1\,\mathsf{then}\,e'_2\,\mathsf{else}\,e'_3)\,\{e_1/x\} \\ (\mathsf{if}\,e'_1\,\mathsf{then}\,e'_2\,\mathsf{else}\,e'_3)\,\{e_2/x\} & \longrightarrow^* & (\mathsf{if}\,e''_1\,\mathsf{then}\,e'_2\,\mathsf{else}\,e'_3)\,\{e_2/x\}.\end{array}$$

Case $e'_1\,\{e_1/x\}$ is a value: Since e'_1 is a value by Lemma 8, we finish by Lemma 14 (1).

Case $e = \langle \{y{:}T \,|\, e'_1\}, e'_2, v\rangle^\ell$: Similarly to the case for application term except for use of Lemma 20 (1). Since $e\,\{e_1/x\}$ takes a step, there are two cases we have to consider.

Case $e'_2\,\{e_1/x\} \longrightarrow e''$ by (E_Red): By the IH, there exists some e''_2 such that $e'_2\,\{e_2/x\} \longrightarrow^* e''_2\,\{e_2/x\}$ and $e'' = e''_2\,\{e_1/x\}$. Moreover, the evaluation $e'_2\,\{e_2/x\} \longrightarrow^* e''_2\,\{e_2/x\}$ is derived by applying only (E_Red). Thus, by Lemma 21 (1),

$$\begin{array}{rcl}(\langle \{y{:}T \,|\, e'_1\}, e'_2, v\rangle^\ell)\,\{e_1/x\} & \longrightarrow & (\langle \{y{:}T \,|\, e'_1\}, e''_2, v\rangle^\ell)\,\{e_1/x\} \\ (\langle \{y{:}T \,|\, e'_1\}, e'_2, v\rangle^\ell)\,\{e_2/x\} & \longrightarrow^* & (\langle \{y{:}T \,|\, e'_1\}, e''_2, v\rangle^\ell)\,\{e_2/x\}.\end{array}$$

Case $e'_2\,\{e_1/x\}$ is a value: Since e'_2 is a value by Lemma 8, we finish by Lemma 20 (1).

Case $e = \langle\!\langle \{y{:}T \,|\, e'_1\}, e'_2\rangle\!\rangle^\ell$: Similarly to the case for application term except for use of Lemma 18 (1). Since $e\,\{e_1/x\}$ takes a step, there are two cases we have to consider.

Case $e'_2\,\{e_1/x\} \longrightarrow e''$ by (E_Red): By the IH, there exists some e''_2 such that $e'_2\,\{e_2/x\} \longrightarrow^* e''_2\,\{e_2/x\}$ and $e'' = e''_2\,\{e_1/x\}$. Moreover, the evaluation $e'_2\,\{e_2/x\} \longrightarrow^* e''_2\,\{e_2/x\}$ is derived by applying only (E_Red). Thus, by Lemma 21 (1),

$$\begin{array}{rcl}(\langle\!\langle \{y{:}T \,|\, e'_1\}, e'_2\rangle\!\rangle^\ell)\,\{e_1/x\} & \longrightarrow & (\langle\!\langle \{y{:}T \,|\, e'_1\}, e''_2\rangle\!\rangle^\ell)\,\{e_1/x\} \\ (\langle\!\langle \{y{:}T \,|\, e'_1\}, e'_2\rangle\!\rangle^\ell)\,\{e_2/x\} & \longrightarrow^* & (\langle\!\langle \{y{:}T \,|\, e'_1\}, e''_2\rangle\!\rangle^\ell)\,\{e_2/x\}.\end{array}$$

Case $e'_2\,\{e_1/x\}$ is a value: Since e'_2 is a value by Lemma 8, we finish by Lemma 18 (1). □

Lemma 24. *If $e_1 \longrightarrow e_2$, and $e\,\{e_2/x\}$ is a value, then there exists some e' such that*

- $e\,\{e_1/x\} \longrightarrow^* e'\,\{e_1/x\}$,
- $e'\,\{e_1/x\}$ *is a value, and*
- $e\,\{e_2/x\} = e'\,\{e_2/x\}$.

Proof. By structural induction on e.

Case $e = y$: If $x = y$, then $e\,\{e_2/x\} = e_2$ is a value. Thus, we finish by letting $e' = e_2$ because $e_2\,\{e_1/x\} = e_2\,\{e_2/x\} = e_2$. Note that e_2 is closed since the evaluation relation is defined over closed terms. Otherwise, if $x \neq y$, then contradiction because $e\,\{e_2/x\}$ is a value but $e\,\{e_2/x\} = y$ is not.

Case $e = v$: Obvious by letting $e' = v$ because $v\,\{e_1/x\}$ is a value by Lemma 7.

Case $e = \Uparrow\ell$, $e'_1\,e'_2$, $e'.i$ for $i \in \{1,2\}$, $\mathsf{match}\; e'_0 \;\mathsf{with}\; \overline{C_i\,y_i \to e'_i}^{\,i}$, $\mathsf{if}\; e'_1 \;\mathsf{then}\; e'_2 \;\mathsf{else}\; e'_3$, $\langle\{y{:}T \mid e'_1\}, e'_2, v\rangle^\ell$ or $\langle\!\langle\{y{:}T \mid e'_1\}, e'_2\rangle\!\rangle^\ell$: Contradictory: $e\,\{e_2/x\}$ is a value.

Case $e = (e'_1, e'_2)$: Let $i \in \{1,2\}$. By the assumption, $e'_i\,\{e_2/x\}$ is a value. By the IH, there exists some e''_i such that $e'_i\,\{e_1/x\} \longrightarrow^* e''_i\,\{e_1/x\}$ and $e''_i\,\{e_1/x\}$ is a value and $e'_i\,\{e_2/x\} = e''_i\,\{e_2/x\}$. Thus, $(e'_1, e'_2)\,\{e_1/x\} \longrightarrow^* (e''_1, e''_2)\,\{e_1/x\}$ and $(e''_1, e''_2)\,\{e_1/x\}$ is a value and $e\,\{e_2/x\} = (e''_1, e''_2)\,\{e_2/x\}$.

Case $e = C\langle e'_1\rangle e'_2$: By the assumption, $e'_2\,\{e_2/x\}$ is a value. By the IH, there exists some e''_2 such that $e'_2\,\{e_1/x\} \longrightarrow^* e''_2\,\{e_1/x\}$ and $e''_2\,\{e_1/x\}$ is a value and $e'_2\,\{e_2/x\} = e''_2\,\{e_2/x\}$. Thus, $(C\langle e'_1\rangle e'_2)\,\{e_1/x\} \longrightarrow^* (C\langle e'_1\rangle e''_2)\,\{e_1/x\}$ and $(C\langle e'_1\rangle e''_2)\,\{e_1/x\}$ is a value and $(C\langle e'_1\rangle e'_2)\,\{e_2/x\} = (C\langle e'_1\rangle e''_2)\,\{e_2/x\}$. □

Lemma 25. *If $e_1 \longrightarrow e_2$ and $e\,\{e_2/x\} = E_2[\Uparrow\ell]$, then $e\,\{e_1/x\} \longrightarrow^* \Uparrow\ell$.*

Proof. By structural induction on e.

Case $e = x$: Obvious since $e_1 \longrightarrow e_2 = e\,\{e_2/x\} = E_2[\Uparrow\ell] \longrightarrow \Uparrow\ell$.

Case $e = \Uparrow\ell$: Obvious.

Case $e = y$ where $y \neq x$, $\Uparrow\ell'$ where $\ell \neq \ell'$, and v: Contradictory (by Lemma 8 in the case that $e = v$) since $e\,\{e_2/x\} = E_2[\Uparrow\ell]$.

Case $e = e'_1\,e'_2$: Since $e\,\{e_2/x\} = E_2[\Uparrow\ell]$, there are two cases we have to consider.

Case $E_2 = E'_2\,e'_2\,\{e_2/x\}$: Since $e'_1\,\{e_2/x\} = E'_2[\Uparrow\ell]$, we have $e'_1\,\{e_1/x\} \longrightarrow^* \Uparrow\ell$ by the IH. Thus, we finish by Lemma 21 (2).

Case $E_2 = e'_1\,\{e_2/x\}\,E'_2$ where $e'_1\,\{e_2/x\}$ is a value: By Lemma 24, there exists some e''_1 such that $e'_1\,\{e_1/x\} \longrightarrow^* e''_1\,\{e_1/x\}$ and $e''_1\,\{e_1/x\}$ is a value and $e'_1\,\{e_2/x\} = e''_1\,\{e_2/x\}$. Since $e'_2\,\{e_2/x\} = E'_2[\Uparrow\ell]$, we have $e'_2\,\{e_1/x\} \longrightarrow^* \Uparrow\ell$ by the IH. Thus, $(e'_1\,e'_2)\,\{e_1/x\} \longrightarrow^* (e''_1\,e'_2)\,\{e_1/x\} \longrightarrow^* \Uparrow\ell$ by Lemmas 21 (1) and (2).

Case $e = (e'_1, e'_2)$: Since $e\,\{e_2/x\} = E_2[\Uparrow\ell]$, there are two cases we have to consider.

Case $E_2 = (E'_2, e'_2\,\{e_2/x\})$: Since $e'_1\,\{e_2/x\} = E'_2[\Uparrow\ell]$, we have $e'_1\,\{e_1/x\} \longrightarrow^* \Uparrow\ell$ by the IH. Thus, we finish by Lemma 21 (2).

Case $E_2 = (e'_1\,\{e_2/x\}, E'_2)$ where $e'_1\,\{e_2/x\}$ is a value: By Lemma 24, there exists some e''_1 such that $e'_1\,\{e_1/x\} \longrightarrow^* e''_1\,\{e_1/x\}$ and $e''_1\,\{e_1/x\}$ is a value and $e'_1\,\{e_2/x\} = e''_1\,\{e_2/x\}$. Since $e'_2\,\{e_2/x\} = E'_2[\Uparrow\ell]$, we have $e'_2\,\{e_1/x\} \longrightarrow^* \Uparrow\ell$ by the IH. Thus, $(e'_1, e'_2)\,\{e_1/x\} \longrightarrow^* (e''_1, e'_2)\,\{e_1/x\} \longrightarrow^* \Uparrow\ell$ by Lemmas 21 (1) and (2).

Case $e = e'.i$ for $i \in \{1,2\}$: Since $e\,\{e_2/x\} = E_2[\Uparrow\ell]$, there exists some E'_2 such that $E_2 = E'_2.i$. Since $e'\,\{e_2/x\} = E'_2[\Uparrow\ell]$, we have $e'\,\{e_1/x\} \longrightarrow^* \Uparrow\ell$ by the IH. By Lemma 21 (2), we finish.

Case $e = C\langle e'_1\rangle e'_2$: Since $e\,\{e_2/x\} = E_2[\Uparrow\ell]$, there exists some E'_2 such that $E_2 = C\langle e'_1\,\{e_2/x\}\rangle E'_2$. Since $e'_2\,\{e_2/x\} = E'_2[\Uparrow\ell]$, we have $e'_2\,\{e_1/x\} \longrightarrow^* \Uparrow\ell$ by the IH. By Lemma 21 (2), we finish.

Case $e = \mathsf{match}\; e'_0 \;\mathsf{with}\; \overline{C_i\,y_i \to e'_i}^{\,i}$: Since $e\,\{e_2/x\} = E_2[\Uparrow\ell]$, there exists some E'_2 such that $E_2 = \mathsf{match}\; E'_2 \;\mathsf{with}\; \overline{C_i\,y_i \to e'_i}^{\,i}$. Since $e'_0\,\{e_2/x\} = E'_2[\Uparrow\ell]$, we have $e'_0\,\{e_1/x\} \longrightarrow^* \Uparrow\ell$ by the IH. By Lemma 21 (2), we finish.

Case $e = \mathsf{if}\; e'_1 \;\mathsf{then}\; e'_2 \;\mathsf{else}\; e'_3$: Since $e\,\{e_2/x\} = E_2[\Uparrow\ell]$, there exists some E'_2 such that $E_2 = \mathsf{if}\; E'_2 \;\mathsf{then}\; e'_2\,\{e_2/x\} \;\mathsf{else}\; e'_3\,\{e_2/x\}$. Since $e'_1\,\{e_2/x\} = E'_2[\Uparrow\ell]$, we have $e'_1\,\{e_1/x\} \longrightarrow^* \Uparrow\ell$ by the IH. By Lemma 21 (2), we finish.

Case $e = \langle\{y{:}T \mid e'_1\}, e'_2, v\rangle^{\ell'}$: Since $e\,\{e_2/x\} = E_2[\Uparrow\ell]$, there exists some E'_2 such that $E_2 = \langle\{y{:}T \mid e'_1\}\,\{e_2/x\}, E'_2, v\,\{e_2/x\}\rangle^{\ell'}$. Since $e'_2\,\{e_2/x\} = E'_2[\Uparrow\ell]$, we have $e'_2\,\{e_1/x\} \longrightarrow^* \Uparrow\ell$ by the IH. By Lemma 21 (2), we finish.

Case $e = \langle\!\langle\{y{:}T \mid e'_1\}, e'_2\rangle\!\rangle^{\ell'}$: Since $e\,\{e_2/x\} = E_2[\Uparrow\ell]$, there exists some E'_2 such that $E_2 = \langle\!\langle\{y{:}T \mid e'_1\}\,\{e_2/x\}, E'_2\rangle\!\rangle^{\ell'}$. Since $e'_2\,\{e_2/x\} = E'_2[\Uparrow\ell]$, we have $e'_2\,\{e_1/x\} \longrightarrow^* \Uparrow\ell$ by the IH. By Lemma 21 (2), we finish. □

Lemma 26 (Cotermination: Reduction on the Right)**.** *Suppose that* $e_1 \longrightarrow e_2$*. If* $e\,\{e_2/x\} \longrightarrow e'$*, then* $e\,\{e_1/x\} \longrightarrow^* e''\,\{e_1/x\}$ *for some* e'' *such that* $e' = e''\,\{e_2/x\}$*. Moreover, if* $e\,\{e_2/x\} \longrightarrow e'$ *is derived by (*E_Red*), then the evaluation* $e\,\{e_1/x\} \longrightarrow^* e''\,\{e_1/x\}$ *is derived by applying only (*E_Red*).*

Proof. By structural induction on e. If $e\,\{e_2/x\} \longrightarrow e'$ is derived by (E_Blame), then there exist some E_2 and ℓ such that $e\,\{e_2/x\} = E_2[\Uparrow\ell]$ and $e' = \Uparrow\ell$. By Lemma 25, $e\,\{e_1/x\} \longrightarrow^* \Uparrow\ell$. We finish by letting $e'' = \Uparrow\ell$.

In what follows, we suppose that $e\,\{e_2/x\}$ is derived by (E_Red). We proceed by case analysis on e.

Case $e = y$: If $x = y$, then we have $e\,\{e_1/x\} = e_1$ and $e\,\{e_2/x\} = e_2$. Thus, we finish by letting $e_1' = e_2'$ because $e_2'\,\{e_1/x\} = e_2'\,\{e_2/x\} = e_2'$. Note that the evaluation relation is defined over closed terms. Otherwise, if $x \neq y$, then contradiction because $e\,\{e_2/x\} = y$ takes a step.

Case $e = \Uparrow\ell$: Contradictory.

Case $e = v$: Contradictory by Lemma 7 since $e\,\{e_2/x\} \longrightarrow e_2'$.

Case $e = e_1'\,e_2'$: Since $e\,\{e_2/x\}$ takes a step, there are three cases we have to consider.

Case $e_1'\,\{e_2/x\} \longrightarrow e''$ by (E_Red): By the IH, there exists some e_1'' such that $e_1'\,\{e_1/x\} \longrightarrow^* e_1''\,\{e_1/x\}$ and $e'' = e_1''\,\{e_2/x\}$. Moreover, the evaluation $e_1'\,\{e_1/x\} \longrightarrow^* e_1''\,\{e_1/x\}$ is derived by applying only (E_Red). Thus, by Lemma 21 (1), $(e_1'\,e_2')\,\{e_2/x\} \longrightarrow (e_1''\,e_2')\,\{e_2/x\}$ and $(e_1'\,e_2')\,\{e_1/x\} \longrightarrow^* (e_1''\,e_2')\,\{e_1/x\}$.

Case $e_1'\,\{e_2/x\}$ is a value and $e_2'\,\{e_2/x\} \longrightarrow e''$ by (E_Red): By Lemma 24, there exists some e_1'' such that $e_1'\,\{e_1/x\} \longrightarrow^* e_1''\,\{e_1/x\}$ and $e_1''\,\{e_1/x\}$ is a value and $e_1'\,\{e_2/x\} = e_1''\,\{e_2/x\}$. By the IH, there exists some e_2'' such that $e_2'\,\{e_1/x\} \longrightarrow^* e_2''\,\{e_1/x\}$ and $e'' = e_2''\,\{e_2/x\}$. Moreover, the evaluation $e_2'\,\{e_1/x\} \longrightarrow^* e_2''\,\{e_1/x\}$ is derived by applying only (E_Red). Thus, by Lemma 21 (1), $(e_1'\,e_2')\,\{e_2/x\} \longrightarrow (e_1''\,e_2'')\,\{e_2/x\}$ and $(e_1'\,e_2')\,\{e_1/x\} \longrightarrow^* (e_1''\,e_2'')\,\{e_1/x\}$.

Case $e_1'\,\{e_2/x\}$ and $e_2'\,\{e_2/x\}$ are values: Let $i \in \{1,2\}$. By Lemma 24, there exist some e_i'' such that $e_i'\,\{e_1/x\} \longrightarrow^* e_i''\,\{e_1/x\}$ and $e_i''\,\{e_1/x\}$ is a value and $e_i'\,\{e_2/x\} = e_i''\,\{e_2/x\}$. Since e_1'' and e_2'' are values by Lemma 8, we finish by Lemmas 10 (2) and 21 (1).

Case $e = (e_1', e_2')$: Similarly to the case for application term. Since $e\,\{e_2/x\}$ takes a step, there are two cases we have to consider.

Case $e_1'\,\{e_2/x\} \longrightarrow e''$ by (E_Red): By the IH, there exists some e_1'' such that $e_1'\,\{e_1/x\} \longrightarrow^* e_1''\,\{e_1/x\}$ and $e'' = e_1''\,\{e_2/x\}$. Moreover, the evaluation $e_1'\,\{e_1/x\} \longrightarrow^* e_1''\,\{e_1/x\}$ is derived by applying only (E_Red). Thus, by Lemma 21 (1), $(e_1', e_2')\,\{e_2/x\} \longrightarrow (e_1'', e_2')\,\{e_2/x\}$ and $(e_1', e_2')\,\{e_1/x\} \longrightarrow^* (e_1'', e_2')\,\{e_1/x\}$.

Case $e_1'\,\{e_2/x\}$ is a value and $e_2'\,\{e_2/x\} \longrightarrow e''$ by (E_Red): By Lemma 24, there exists some e_1'' such that $e_1'\,\{e_1/x\} \longrightarrow^* e_1''\,\{e_1/x\}$ and $e_1''\,\{e_1/x\}$ is a value and $e_1'\,\{e_2/x\} = e_1''\,\{e_2/x\}$. By the IH, there exists some e_2'' such that $e_2'\,\{e_1/x\} \longrightarrow^* e_2''\,\{e_1/x\}$ and $e'' = e_2''\,\{e_2/x\}$. Moreover, the evaluation $e_2'\,\{e_1/x\} \longrightarrow^* e_2''\,\{e_1/x\}$ is derived by applying only (E_Red). Thus, by Lemma 21 (1), $(e_1', e_2')\,\{e_2/x\} \longrightarrow (e_1'', e_2'')\,\{e_2/x\}$ and $(e_1', e_2')\,\{e_1/x\} \longrightarrow^* (e_1'', e_2'')\,\{e_1/x\}$.

Case $e = e'.i$ for $i \in \{1,2\}$: Similarly to the case for application term except for use of Lemma 12 (2). If there exists some e'' such that $e'\,\{e_2/x\} \longrightarrow e''$ by (E_Red), then, by the IH, there exists some e''' such that $e'\,\{e_1/x\} \longrightarrow^* e'''\,\{e_1/x\}$ and $e'' = e'''\,\{e_2/x\}$. Moreover, the evaluation $e'\,\{e_1/x\} \longrightarrow^* e'''\,\{e_1/x\}$ is derived by applying only (E_Red). Thus, by Lemma 21 (1), $(e'.i)\,\{e_2/x\} \longrightarrow (e'''.i)\,\{e_2/x\}$ and $(e'.i)\,\{e_1/x\} \longrightarrow^* (e'''.i)\,\{e_1/x\}$. Otherwise, if $e'\,\{e_2/x\}$ is a value, then there exists some e'' such that $e'\,\{e_1/x\} \longrightarrow^* e''\,\{e_1/x\}$ and $e''\,\{e_1/x\}$ is a value and $e'\,\{e_2/x\} = e''\,\{e_2/x\}$. Since e'' is a value by Lemma 8, we finish by Lemmas 12 (2) and 21 (1).

Case $e = C\langle e_1'\rangle e_2'$: Similarly to the case for application term. Since $e\,\{e_2/x\}$ takes a step, there exists some e'' such that $e_2'\,\{e_2/x\} \longrightarrow e''$ by (E_Red). By the IH, there exists some e_2'' such that $e_2'\,\{e_1/x\} \longrightarrow^* e_2''\,\{e_1/x\}$ and $e'' = e_2''\,\{e_2/x\}$. Moreover, the evaluation $e_2'\,\{e_1/x\} \longrightarrow^* e_2''\,\{e_1/x\}$ is derived by applying only (E_Red). Thus, by Lemma 21 (1), $(C\langle e_1'\rangle e_2')\,\{e_2/x\} \longrightarrow (C\langle e_1'\rangle e_2'')\,\{e_2/x\}$ and $(C\langle e_1'\rangle e_2')\,\{e_1/x\} \longrightarrow^* (C\langle e_1'\rangle e_2'')\,\{e_1/x\}$.

Case $e = \mathsf{match}\, e'_0\, \mathsf{with}\, \overline{C_i\, y_i \to e'_i}^{\,i}$: Similarly to the case for application term except for use of Lemma 16 (2). If there exists some e'' such that $e'_0\,\{e_2/x\} \longrightarrow e''$ by (E_Red), then, by the IH, there exists some e''_0 such that $e'_0\,\{e_1/x\} \longrightarrow^* e''_0\,\{e_1/x\}$ and $e'' = e''_0\,\{e_2/x\}$. Moreover, the evaluation $e'_0\,\{e_1/x\} \longrightarrow^* e''_0\,\{e_1/x\}$ is derived by applying only (E_Red). Thus, by Lemma 21 (1),

$$
\begin{array}{lcl}
(\mathsf{match}\, e'_0\, \mathsf{with}\, \overline{C_i\, y_i \to e'_i}^{\,i})\,\{e_2/x\} & \longrightarrow & (\mathsf{match}\, e''_0\, \mathsf{with}\, \overline{C_i\, y_i \to e'_i}^{\,i})\,\{e_2/x\} \\
(\mathsf{match}\, e'_0\, \mathsf{with}\, \overline{C_i\, y_i \to e'_i}^{\,i})\,\{e_1/x\} & \longrightarrow^* & (\mathsf{match}\, e''_0\, \mathsf{with}\, \overline{C_i\, y_i \to e'_i}^{\,i})\,\{e_1/x\}.
\end{array}
$$

Otherwise, if $e'_0\,\{e_2/x\}$ is a value, then there exists some e''_0 such that $e'_0\,\{e_1/x\} \longrightarrow^* e''_0\,\{e_1/x\}$ and $e''_0\,\{e_1/x\}$ is a value and $e'_0\,\{e_2/x\} = e''_0\,\{e_2/x\}$. Since e''_0 is a value by Lemma 8, we finish by Lemmas 16 (2) and 21 (1).

Case $e = \mathsf{if}\, e'_1\, \mathsf{then}\, e'_2\, \mathsf{else}\, e'_3$: Similarly to the case for application term except for use of Lemma 14 (2). If there exists some e'' such that $e'_1\,\{e_2/x\} \longrightarrow e''$ by (E_Red), then, by the IH, there exists some e''_1 such that $e'_1\,\{e_1/x\} \longrightarrow^* e''_1\,\{e_1/x\}$ and $e'' = e''_1\,\{e_2/x\}$. Moreover, the evaluation $e'_1\,\{e_1/x\} \longrightarrow^* e''_1\,\{e_1/x\}$ is derived by applying only (E_Red). Thus, by Lemma 21 (1),

$$
\begin{array}{lcl}
(\mathsf{if}\, e'_1\, \mathsf{then}\, e'_2\, \mathsf{else}\, e'_3)\,\{e_2/x\} & \longrightarrow & (\mathsf{if}\, e''_1\, \mathsf{then}\, e'_2\, \mathsf{else}\, e'_3)\,\{e_2/x\} \\
(\mathsf{if}\, e'_1\, \mathsf{then}\, e'_2\, \mathsf{else}\, e'_3)\,\{e_1/x\} & \longrightarrow^* & (\mathsf{if}\, e''_1\, \mathsf{then}\, e'_2\, \mathsf{else}\, e'_3)\,\{e_1/x\}.
\end{array}
$$

Otherwise, if $e'_1\,\{e_2/x\}$ is a value, then there exists some e''_1 such that $e'_1\,\{e_1/x\} \longrightarrow^* e''_1\,\{e_1/x\}$ and $e''_1\,\{e_1/x\}$ is a value and $e'_1\,\{e_2/x\} = e''_1\,\{e_2/x\}$. Since e''_1 is a value by Lemma 8, we finish by Lemmas 14 (2) and 21 (1).

Case $e = \langle\{y{:}T \mid e'_1\}, e'_2, v\rangle^{\ell}$: Similarly to the case for application term except for use of Lemma 20 (2). If there exists some e'' such that $e'_2\,\{e_2/x\} \longrightarrow e''$ by (E_Red), then, by the IH, there exists some e''_2 such that $e'_2\,\{e_1/x\} \longrightarrow^* e''_2\,\{e_1/x\}$ and $e'' = e''_2\,\{e_2/x\}$. Moreover, the evaluation $e'_2\,\{e_1/x\} \longrightarrow^* e''_2\,\{e_1/x\}$ is derived by applying only (E_Red). Thus, by Lemma 21 (1),

$$
\begin{array}{lcl}
(\langle\{y{:}T \mid e'_1\}, e'_2, v\rangle^{\ell})\,\{e_2/x\} & \longrightarrow & (\langle\{y{:}T \mid e'_1\}, e''_2, v\rangle^{\ell})\,\{e_2/x\} \\
(\langle\{y{:}T \mid e'_1\}, e'_2, v\rangle^{\ell})\,\{e_1/x\} & \longrightarrow^* & (\langle\{y{:}T \mid e'_1\}, e''_2, v\rangle^{\ell})\,\{e_1/x\}.
\end{array}
$$

Otherwise, if $e'_2\,\{e_2/x\}$ is a value, then there exists some e''_2 such that $e'_2\,\{e_1/x\} \longrightarrow^* e''_2\,\{e_1/x\}$ and $e''_2\,\{e_1/x\}$ is a value and $e'_2\,\{e_2/x\} = e''_2\,\{e_2/x\}$. Since e''_2 is a value by Lemma 8, we finish by Lemmas 20 (2) and 21 (1).

Case $e = \langle\!\langle\{y{:}T \mid e'_1\}, e'_2\rangle\!\rangle^{\ell}$: Similarly to the case for application term except for use of Lemma 18 (2). If there exists some e'' such that $e'_2\,\{e_2/x\} \longrightarrow e''$ by (E_Red), then, by the IH, there exists some e''_2 such that $e'_2\,\{e_1/x\} \longrightarrow^* e''_2\,\{e_1/x\}$ and $e'' = e''_2\,\{e_2/x\}$. Moreover, the evaluation $e'_2\,\{e_1/x\} \longrightarrow^* e''_2\,\{e_1/x\}$ is derived by applying only (E_Red). Thus, by Lemma 21 (1),

$$
\begin{array}{lcl}
(\langle\!\langle\{y{:}T \mid e'_1\}, e'_2\rangle\!\rangle^{\ell})\,\{e_2/x\} & \longrightarrow & (\langle\!\langle\{y{:}T \mid e'_1\}, e''_2\rangle\!\rangle^{\ell})\,\{e_2/x\} \\
(\langle\!\langle\{y{:}T \mid e'_1\}, e'_2\rangle\!\rangle^{\ell})\,\{e_1/x\} & \longrightarrow^* & (\langle\!\langle\{y{:}T \mid e'_1\}, e''_2\rangle\!\rangle^{\ell})\,\{e_1/x\}.
\end{array}
$$

Otherwise, if $e'_2\,\{e_2/x\}$ is a value, then there exists some e''_2 such that $e'_2\,\{e_1/x\} \longrightarrow^* e''_2\,\{e_1/x\}$ and $e''_2\,\{e_1/x\}$ is a value and $e'_2\,\{e_2/x\} = e''_2\,\{e_2/x\}$. Since e''_2 is a value by Lemma 8, we finish by Lemmas 18 (2) and 21 (1). $\square$

Lemma 27. *Suppose that* $e_1 \longrightarrow e_2$.

(1) If $e\,\{e_1/x\} \longrightarrow^* v_1$, *then* $e\,\{e_2/x\} \longrightarrow^* e'\,\{e_2/x\}$ *for some* e' *such that* $v_1 = e'\,\{e_1/x\}$, *and* $e'\,\{e_2/x\}$ *is a value.*

(2) If $e\,\{e_2/x\} \longrightarrow^* v_2$, *then* $e\,\{e_1/x\} \longrightarrow^* e'\,\{e_1/x\}$ *for some* e' *such that* $v_2 = e'\,\{e_2/x\}$, *and* $e'\,\{e_1/x\}$ *is a value.*

Proof.

1. By mathematical induction on the number of evaluation steps of $e\,\{e_1/x\}$.

Case 0: We are given $e\{e_1/x\}$ is a value. Since e_1 is not a value from $e_1 \longrightarrow e_2$, we find that e is a value by Lemma 8. By Lemma 7, so is $e\{e_2/x\}$. Thus, we finish when letting $e' = e$.

Case $i+1$: We are given $e\{e_1/x\} \longrightarrow e'_1 \longrightarrow^i v_1$. By Lemma 23, there exists some e'' such that $e\{e_2/x\} \longrightarrow^* e''\{e_2/x\}$ and $e'_1 = e''\{e_1/x\}$. By the IH, there exists some e' such that $e''\{e_2/x\} \longrightarrow^* e'\{e_2/x\}$ and $v_1 = e'\{e_1/x\}$, and $e'\{e_2/x\}$ is a value. Thus, we finish.

2. By mathematical induction on the number of evaluation steps of $e\{e_2/x\}$.

Case 0: We are given $e\{e_2/x\}$ is a value. By Lemma 24, there exists some e' such that $e\{e_1/x\} \longrightarrow^* e'\{e_1/x\}$ and $e\{e_2/x\} = e'\{e_2/x\}$ and $e'\{e_1/x\}$ is a value.

Case $i+1$: We are given $e\{e_2/x\} \longrightarrow e'_2 \longrightarrow^i v_2$. By Lemma 26, there exists some e'' such that $e\{e_1/x\} \longrightarrow^* e''\{e_1/x\}$ and $e'_2 = e''\{e_2/x\}$. By the IH, there exists some e' such that $e''\{e_1/x\} \longrightarrow^* e'\{e_1/x\}$ and $v_2 = e'\{e_2/x\}$, and $e'\{e_1/x\}$ is a value. Thus, we finish. □

Lemma 28. *Suppose that $e_1 \Rrightarrow^* e_2$.*

(1) If $e_1 \longrightarrow^* v_1$, then $e_2 \longrightarrow^* v_2$ for some v_2 such that $v_1 \Rrightarrow^* v_2$.

(2) If $e_2 \longrightarrow^* v_2$, then $e_1 \longrightarrow^* v_1$ for some v_1 such that $v_1 \Rrightarrow^* v_2$.

Proof. By mathematical induction on the number of steps of $e_1 \Rrightarrow^* e_2$.

Case 0: Obvious because $e_1 = e_2$.

Case $i+1$: We are given $e_1 \Rrightarrow e_3 \Rrightarrow^i e_2$. We are given some e, e'_1, e'_3 and x such that $e_1 = e\{e'_1/x\}$ and $e_3 = e\{e'_3/x\}$ and $e'_1 \longrightarrow e'_3$. Thus, we finish by Lemma 27 and the IHs and transitivity of $\Rrightarrow^*$. □

Lemma 29.

(1) If $c \Rrightarrow^* v$, then $v = c$.

(2) If $v \Rrightarrow^* c$, then $v = c$.

Proof.

1. By mathematical induction on the number of steps of $c \Rrightarrow^* v$.

Case 0: Obvious.

Case $i+1$: We are given $c \Rrightarrow e \Rrightarrow^* v$. We are given e', e_1, e_2 and x such that $c = e'\{e_1/x\}$ and $e = e'\{e_2/x\}$ and $e_1 \longrightarrow e_2$. Since e_1 is not a value from $e_1 \longrightarrow e_2$, we find that e' is a value by Lemma 8. Thus, $e' = c$ and so $e = c$. By the IH, we finish.

2. By mathematical induction on the number of steps of $v \Rrightarrow^* c$.

Case 0: Obvious.

Case $i+1$: We are given $v \Rrightarrow e \Rrightarrow^* c$. We are given e', e_1, e_2 and x such that $v = e'\{e_1/x\}$ and $e = e'\{e_2/x\}$ and $e_1 \longrightarrow e_2$. Since e_1 is not a value from $e_1 \longrightarrow e_2$, we find that e' is a value by Lemma 8. Thus, so is $e'\{e_2/x\}$ by Lemma 7. By the IH, $e'\{e_2/x\} = c$. Since e' is a value, $e' = c$ and so $v = c$. □

Lemma 30 (Cotermination at true)**.** *Suppose that $e_1 \Rrightarrow^* e_2$.*

(1) If $e_1 \longrightarrow^* \mathsf{true}$, then $e_2 \longrightarrow^* \mathsf{true}$.

(2) If $e_2 \longrightarrow^* \mathsf{true}$, then $e_1 \longrightarrow^* \mathsf{true}$.

Proof. By Lemmas 28 and 29. □

Lemma 31. *Suppose that $e_1 \equiv e_2$.*

(1) If $e_1 \longrightarrow^* \mathsf{true}$, then $e_2 \longrightarrow^* \mathsf{true}$.

(2) If $e_2 \longrightarrow^* \mathsf{true}$, then $e_1 \longrightarrow^* \mathsf{true}$.

Proof. Straightforward by induction on $e_1 \equiv e_2$. In particular, if $e_1 \Rrightarrow e_2$, then we finish by Lemma 30. □

4 Type Soundness

Lemma 32 (Weakening)**.** *Suppose that x is a fresh variable and $\Gamma_1 \vdash T_1$.*

(1) If $\Gamma_1, \Gamma_2 \vdash e : T$, then $\Gamma_1, x{:}T_1, \Gamma_2 \vdash e : T$.

(2) If $\Gamma_1, \Gamma_2 \vdash T$, then $\Gamma_1, x{:}T_1, \Gamma_2 \vdash T$.

(3) If $\vdash \Gamma_1, \Gamma_2$, then $\vdash \Gamma_1, x{:}T_1, \Gamma_2$.

Proof. Straightforward by induction on each derivation. □

Lemma 33 (Substitution)**.** *Suppose that $\Gamma_1 \vdash e' : T'$.*

(1) If $\Gamma_1, x{:}T', \Gamma_2 \vdash e : T$, then $\Gamma_1, \Gamma_2\,\{e'/x\} \vdash e\,\{e'/x\} : T\,\{e'/x\}$.

(2) If $\Gamma_1, x{:}T', \Gamma_2 \vdash T$, then $\Gamma_1, \Gamma_2\,\{e'/x\} \vdash T\,\{e'/x\}$.

(3) If $\vdash \Gamma_1, x{:}T', \Gamma_2$, then $\vdash \Gamma_1, \Gamma_2\,\{e'/x\}$.

Proof. Straightforward by induction on each derivation. The only interesting cases are for (T_Ctr) and (T_Match).

Case (T_Ctr): We are given $\Gamma_1, x{:}T', \Gamma_2 \vdash C\langle e_1\rangle e_2 : \tau\langle e_1\rangle$ for some C, e_1, e_2 and τ. By inversion, we have $\mathit{TypSpecOf}(C) = y{:}T_1 \twoheadrightarrow T_2 \twoheadrightarrow \tau\langle y\rangle$ and $\Gamma_1, x{:}T', \Gamma_2 \vdash e_1 : T_1$ and $\Gamma_1, x{:}T', \Gamma_2 \vdash e_2 : T_2\,\{e_1/y\}$ and $\Gamma_1, x{:}T', \Gamma_2 \vdash \tau\langle e_1\rangle$. Without loss of generality, we can suppose that y is fresh.

By the IHs, $\Gamma_1, \Gamma_2\,\{e'/x\} \vdash e_1\,\{e'/x\} : T_1\,\{e'/x\}$ and $\Gamma_1, \Gamma_2\,\{e'/x\} \vdash e_2\,\{e'/x\} : T_2\,\{e_1/y\}\,\{e'/x\}$ and $\Gamma_1, \Gamma_2\,\{e'/x\} \vdash \tau\langle e_1\,\{e'/x\}\rangle$. From well-formedness of the type definition environment, it is found that $T_1\,\{e'/x\} = T_1$ and $T_2\,\{e_1/y\}\,\{e'/x\} = T_2\,\{e_1\,\{e'/x\}/y\}$. Thus, we finish by (T_Ctr).

Case (T_Match): We are given $\Gamma_1, x{:}T', \Gamma_2 \vdash \mathsf{match}\, e_0\, \mathsf{with}\, \overline{C_i\, y_i \rightarrow e_i}^{\,i} : T$. By inversion, we have $\Gamma_1, x{:}T', \Gamma_2 \vdash e_0 : \tau\langle e''\rangle$ and $\Gamma_1, x{:}T', \Gamma_2 \vdash T$ and $\mathit{CtrsOf}(\tau) = \overline{C_i}^{\,i}$ and $\mathit{ArgTypeOf}(\tau) = z{:}T''$ and, for all i, $\mathit{CtrArgOf}(C_i) = T_i$ and $\Gamma_1, x{:}T', \Gamma_2, y_i{:}T_i\,\{e''/z\} \vdash e_i : T$. Without loss of generality, we can suppose that $\overline{y_i}^{\,i}$ and z are fresh.

By the IHs, $\Gamma_1, \Gamma_2\,\{e'/x\} \vdash e_0\,\{e'/x\} : \tau\langle e''\,\{e'/x\}\rangle$ and $\Gamma_1, \Gamma_2\,\{e'/x\} \vdash T\,\{e'/x\}$ and $\Gamma_1, \Gamma_2\,\{e'/x\}, y_i{:}T_i\,\{e''/z\}\,\{e'/x\} \vdash e_i\,\{e'/x\} : T\,\{e'/x\}$. From well-formedness of the type definition environment, it is found that $T_i\,\{e''/z\}\,\{e'/x\} = T_i\,\{e''\,\{e'/x\}/z\}$. Thus, we finish by (T_Match). □

Lemma 34 (Base Types Equivalence Inversion)**.** *If $T_1 \equiv T_2$, then*

(1) $T_1 = \mathsf{Bool}$ implies $T_2 = \mathsf{Bool}$, and

(2) $T_2 = \mathsf{Bool}$ implies $T_1 = \mathsf{Bool}$.

Proof. Straightforward by induction on $T_1 \equiv T_2$. In particular, if $T_1 \Rrightarrow T_2$, then there exist some T, x, e_1 and e_2 such that $T_1 = T\,\{e_1/x\}$ and $T_2 = T\,\{e_2/x\}$. Since $T_1 = \mathsf{Bool}$ or $T_2 = \mathsf{Bool}$, we have $T = \mathsf{Bool}$. Thus $T_1 = T_2 = \mathsf{Bool}$. □

Lemma 35 (Dependent Function Types Equivalence Inversion)**.** *If $T_1 \equiv T_2$, then*

(1) $T_1 = x{:}T_{11} \rightarrow T_{12}$ implies

- *$T = x{:}T_{21} \rightarrow T_{22}$,*
- *$T_{11} \equiv T_{21}$, and*
- *$T_{12} \equiv T_{22}$*

for some T_{21} and T_{22}, and

(2) $T_2 = x{:}T_{21} \rightarrow T_{22}$ implies

- *$T_1 = x{:}T_{11} \rightarrow T_{12}$,*
- *$T_{11} \equiv T_{21}$, and*

- $T_{12} \equiv T_{22}$

for some T_{11} and T_{12}.

Proof. Straightforward by induction on $T_1 \equiv T_2$. In particular, if $T_1 \Rrightarrow T_2$, then there exist some T, y, e_1 and e_2 such that $T_1 = T\{e_1/y\}$ and $T_2 = T\{e_2/y\}$ and $e_1 \longrightarrow e_2$. Without loss of generality, we can suppose that x is fresh for e_1, e_2 and y. Since $T_1 = x{:}T_{11} \to T_{12}$ or $T_2 = x{:}T_{21} \to T_{22}$, we have $T = x{:}T_1 \to T_2$ for some T_1 and T_2. Thus, $T_1 = x{:}T_1\{e_1/y\} \to T_2\{e_1/y\}$ and $T_2 = x{:}T_1\{e_2/y\} \to T_2\{e_2/y\}$. We have $T_1\{e_1/y\} \Rrightarrow T_1\{e_2/y\}$ and $T_2\{e_1/y\} \Rrightarrow T_2\{e_2/y\}$ by definition. □

Lemma 36 (Dependent Product Types Equivalence Inversion). *If $T_1 \equiv T_2$, then*

(1) $T_1 = x{:}T_{11} \times T_{12}$ implies

- $T_2 = x{:}T_{21} \times T_{22}$,
- $T_{11} \equiv T_{21}$, *and*
- $T_{12} \equiv T_{22}$

for some T_{21} and T_{22}, and

(2) $T_2 \equiv x{:}T_{21} \times T_{22}$ implies

- $T_1 = x{:}T_{11} \times T_{12}$,
- $T_{11} \equiv T_{21}$, *and*
- $T_{12} \equiv T_{22}$.

for some T_{11} and T_{12}.

Proof. Similarly to Lemma 35, straightforward by induction on $T_1 \equiv T_2$. In particular, if $T_1 \Rrightarrow T_2$, then there exist some T, y, e_1 and e_2 such that $T_1 = T\{e_1/y\}$ and $T_2 = T\{e_2/y\}$ and $e_1 \longrightarrow e_2$. Without loss of generality, we can suppose that x is fresh for e_1, e_2 and y. Since $T_1 = x{:}T_{11} \times T_{12}$ or $T_2 = x{:}T_{21} \times T_{22}$, we have $T = x{:}T_1 \times T_2$ for some T_1 and T_2. Thus, $T_1 = x{:}T_1\{e_1/y\} \times T_2\{e_1/y\}$ and $T_2 = x{:}T_1\{e_2/y\} \times T_2\{e_2/y\}$. We have $T_1\{e_1/y\} \Rrightarrow T_1\{e_2/y\}$ and $T_2\{e_1/y\} \Rrightarrow T_2\{e_2/y\}$ by definition. □

Lemma 37 (Datatypes Equivalence Inversion). *If $T_1 \equiv T_2$, then*

(1) $T_1 = \tau\langle e_1 \rangle$ implies $T_2 = \tau\langle e_2 \rangle$ and $e_1 \equiv e_2$ for some e_2, and

(2) $T_2 = \tau\langle e_2 \rangle$ implies $T_1 = \tau\langle e_1 \rangle$ and $e_1 \equiv e_2$ for some e_1.

Proof. Similarly to Lemma 35, straightforward by induction on $T_1 \equiv T_2$. In particular, if $T_1 \Rrightarrow T_2$, then there exist some T, x, e'_1 and e'_2 such that $T_1 = T\{e'_1/x\}$ and $T_2 = T\{e'_2/x\}$ and $e'_1 \longrightarrow e'_2$. Since $T_1 = \tau\langle e_1 \rangle$ or $T_2 = \tau\langle e_2 \rangle$, we have $T = \tau\langle e \rangle$ for some e. Thus, $T_1 = \tau\langle e\{e'_1/x\} \rangle$ and $T_2 = \tau\langle e\{e'_2/x\} \rangle$. We have $e\{e'_1/x\} \Rrightarrow e\{e'_2/x\}$ by definition. □

Lemma 38 (Refinement Types Equivalence Inversion). *If $T_1 \equiv T_2$, then*

(1) $T_1 = \{x{:}T'_1 \mid e'_1\}$ implies

- $T_2 = \{x{:}T'_2 \mid e'_2\}$,
- $T'_1 \equiv T'_2$, *and*
- $e'_1 \equiv e'_2$

for some T'_2 and e'_2, and

(2) $T_2 = \{x{:}T'_2 \mid e'_2\}$ implies

- $T_1 = \{x{:}T'_1 \mid e'_1\}$,
- $T'_1 \equiv T'_2$, *and*

- $e'_1 \equiv e'_2$

for some T'_1 and e'_1.

Proof. Similarly to Lemma 35, straightforward by induction on $T_1 \equiv T_2$. In particular, if $T_1 \Rrightarrow T_2$, then there exist some T, y, e''_1 and e''_2 such that $T_1 = T\,\{e''_1/y\}$ and $T_2 = T\,\{e''_2/y\}$ and $e''_1 \longrightarrow e''_2$. Without loss of generality, we can suppose that x is fresh for e''_1, e''_2 and y. Since $T_1 = \{x{:}T'_1 \mid e'_1\}$ or $T_2 = \{x{:}T'_2 \mid e'_2\}$, we have $T = \{x{:}T' \mid e'\}$ for some T' and e'. Thus, $T_1 = \{x{:}T'\,\{e''_1/y\} \mid e'\,\{e''_1/y\}\}$ and $T_2 = \{x{:}T'\,\{e''_2/y\} \mid e'\,\{e''_2/y\}\}$. We have $T'\,\{e''_1/y\} \Rrightarrow T'\,\{e''_2/y\}$ and $e'\,\{e''_1/y\} \Rrightarrow e'\,\{e''_2/y\}$ by definition. □

Lemma 39 (Type Equivalence Closed Under Unrefine)**.** *If $T_1 \equiv T_2$, then $unref(T_1) \equiv unref(T_2)$.*

Proof. By induction on T_1.

Case $T_1 = \mathsf{Bool}$, $x{:}T'_1 \to T'_2$, $x{:}T'_1 \times T'_2$, or $\tau\langle e\rangle$: We have $unref(T_1) = T_1$. Since $T_1 \equiv T_2$, we find that $unref(T_2) = T_2$ by Lemmas 34 (1), 35 (1), 36 (1) and 37 (1). Thus, we finish.

Case $T_1 = \{x{:}T'_1 \mid e'_1\}$: By Lemma 38 (1), there exist some T'_2 and e'_2 such that $T_2 = \{x{:}T'_2 \mid e'_2\}$ and $T'_1 \equiv T'_2$. By the IH, $unref(T'_1) \equiv unref(T'_2)$. Because $unref(T_1) = unref(T'_1)$ and $unref(T_2) = unref(T'_2)$, we finish.

□

Lemma 40 (Lambda Inversion)**.** *If $\Gamma \vdash \mathtt{fix}\ f(x{:}T_1){:}T_2 = e : T$, then*

- *$\Gamma, f{:}(x{:}T_1 \to T_2), x{:}T_1 \vdash e : T_2$,*
- *$f \notin \mathsf{FV}(T_2)$, and*
- *$x{:}T_1 \to T_2 \equiv unref(T)$.*

Proof. By induction on the typing derivation. Only four rules can be applied to the lambda abstraction.

Case (T_Abs): Since $T = x{:}T_1 \to T_2$, we have $x{:}T_1 \to T_2 \equiv unref(T)$ by Lemma 1 (reflexivity). By inversion, we finish.

Case (T_Conv): By inversion, we have $\varnothing \vdash \mathtt{fix}\ f(x{:}T_1){:}T_2 = e : T'$ and $T' \equiv T$ for some T'. By the IH, we have $f{:}(x{:}T_1 \to T_2), x{:}T_1 \vdash e : T_2$ and $f \notin \mathsf{FV}(T_2)$ and $x{:}T_1 \to T_2 \equiv unref(T')$. Because $unref(T') \equiv unref(T)$ by Lemma 39, we have $x{:}T_1 \to T_2 \equiv unref(T)$ by Lemma 1 (transitivity). By Lemma 32, we finish.

Case (T_Forget): By inversion, we have $\varnothing \vdash \mathtt{fix}\ f(x{:}T_1){:}T_2 = e : \{y{:}T \mid e'\}$ for some y and e'. By the IH, $f{:}(x{:}T_1 \to T_2), x{:}T_1 \vdash e : T_2$ and $f \notin \mathsf{FV}(T_2)$ and $x{:}T_1 \to T_2 \equiv unref(\{y{:}T \mid e'\})$. Since $unref(T) = unref(\{y{:}T \mid e'\})$, we have $x{:}T_1 \to T_2 \equiv unref(T)$. By Lemma 32, we finish.

Case (T_Exact): We are given $\Gamma \vdash \mathtt{fix}\ f(x{:}T_1){:}T_2 = e : \{y{:}T' \mid e'\}$ for some y, T' and e'. By inversion, we have $\varnothing \vdash \mathtt{fix}\ f(x{:}T_1){:}T_2 = e : T'$. By the IH, we have $f{:}(x{:}T_1 \to T_2), x{:}T_1 \vdash e : T_2$ and $f \notin \mathsf{FV}(T_2)$ and $x{:}T_1 \to T_2 \equiv unref(T')$. Since $unref(T') = unref(\{y{:}T' \mid e'\})$, we have $x{:}T_1 \to T_2 \equiv unref(\{y{:}T' \mid e'\})$. By Lemma 32, we finish. □

Lemma 41 (Cast Inversion)**.** *If $\Gamma \vdash \langle T_1 \Leftarrow T_2\rangle^{\ell} : T$, then*

- *$\Gamma \vdash T_1$,*
- *$\Gamma \vdash T_2$,*
- *$T_1 \parallel T_2$, and*
- *$T_2 \to T_1 \equiv unref(T)$.*

Proof. Similarly to Lemma 40, by induction on the typing derivation. Only four rules can be applied to the cast.

Case (T_Cast): Since $T = T_2 \to T_1$, we have $T_2 \to T_1 \equiv unref(T)$ by Lemma 1 (reflexivity). By inversion, we finish.

Case (T_Conv): By inversion, we have $\varnothing \vdash \langle T_1 \Leftarrow T_2\rangle^\ell : T'$ and $T' \equiv T$ for some T'. By the IH, we have $\varnothing \vdash T_1$ and $\varnothing \vdash T_2$ and $T_1 \parallel T_2$ and $T_2 \to T_1 \equiv \mathit{unref}(T')$. Because $\mathit{unref}(T') \equiv \mathit{unref}(T)$ by Lemma 39, we have $T_2 \to T_1 \equiv \mathit{unref}(T)$ by Lemma 1 (transitivity). By Lemma 32, we finish.

Case (T_Forget): By inversion, we have $\varnothing \vdash \langle T_1 \Leftarrow T_2\rangle^\ell : \{y{:}T \mid e\}$ for some y and e. By the IH, $\varnothing \vdash T_1$ and $\varnothing \vdash T_2$ and $T_1 \parallel T_2$ and $T_2 \to T_1 \equiv \mathit{unref}(\{y{:}T \mid e\})$. Since $\mathit{unref}(\{y{:}T \mid e\}) = \mathit{unref}(T)$, we have $T_2 \to T_1 \equiv \mathit{unref}(T)$. By Lemma 32, we finish.

Case (T_Exact): We are given $\Gamma \vdash \langle T_1 \Leftarrow T_2\rangle^\ell : \{x{:}T' \mid e'\}$ for some x, T' and e'. By inversion, we have $\varnothing \vdash \langle T_1 \Leftarrow T_2\rangle^\ell : T'$. By the IH, we have $\varnothing \vdash T_1$ and $\varnothing \vdash T_2$ and $T_1 \parallel T_2$ and $T_2 \to T_1 \equiv \mathit{unref}(T')$. Since $\mathit{unref}(T') = \mathit{unref}(\{x{:}T' \mid e'\})$, we have $T_2 \to T_1 \equiv \mathit{unref}(\{x{:}T' \mid e'\})$. By Lemma 32, we finish. ☐

Lemma 42 (Pair Inversion)**.** *If* $\Gamma \vdash (v_1, v_2) : T$, *then*

- $\Gamma \vdash v_1 : T_1$,
- $\Gamma \vdash v_2 : T_2\,\{v_1/x\}$,
- $\Gamma, x{:}T_1 \vdash T_2$, *and*
- $x{:}T_1 \times T_2 \equiv \mathit{unref}(T)$

for some T_1, T_2 *and* x.

Proof. Similarly to Lemma 40, by induction on the typing derivation. Only four rules can be applied to the pair.

Case (T_Pair): Since $T = x{:}T_1 \times T_2$, we have $x{:}T_1 \times T_2 \equiv \mathit{unref}(T)$ by Lemma 1 (reflexivity). By inversion, we finish.

Case (T_Conv): By inversion, we have $\varnothing \vdash (v_1, v_2) : T'$ and $T' \equiv T$ for some T'. By the IH, we have $\varnothing \vdash v_1 : T_1$ and $\varnothing \vdash v_2 : T_2\,\{v_1/x\}$ and $x{:}T_1 \vdash T_2$ and $x{:}T_1 \times T_2 \equiv \mathit{unref}(T')$. Because $\mathit{unref}(T') \equiv \mathit{unref}(T)$ by Lemma 39, we have $x{:}T_1 \times T_2 \equiv \mathit{unref}(T)$ by Lemma 1 (transitivity). By Lemma 32, we finish.

Case (T_Forget): By inversion, we have $\varnothing \vdash (v_1, v_2) : \{y{:}T \mid e'\}$ for some y and e'. By the IH, we have $\varnothing \vdash v_1 : T_1$ and $\varnothing \vdash v_2 : T_2\,\{v_1/x\}$ and $x{:}T_1 \vdash T_2$ and $x{:}T_1 \times T_2 \equiv \mathit{unref}(\{y{:}T \mid e'\})$. Since $\mathit{unref}(\{y{:}T \mid e'\}) = \mathit{unref}(T)$, we have $x{:}T_1 \times T_2 \equiv \mathit{unref}(T)$. By Lemma 32, we finish.

Case (T_Exact): We are given $\Gamma \vdash (v_1, v_2) : \{y{:}T' \mid e'\}$ for some y, T' and e'. By inversion, we have $\varnothing \vdash (v_1, v_2) : T'$. By the IH, we have $\varnothing \vdash v_1 : T_1$ and $\varnothing \vdash v_2 : T_2\,\{v_1/x\}$ and $x{:}T_1 \vdash T_2$ and $x{:}T_1 \times T_2 \equiv \mathit{unref}(T')$. Since $\mathit{unref}(T') = \mathit{unref}(\{y{:}T' \mid e'\})$, we have $x{:}T_1 \times T_2 \equiv \mathit{unref}(\{y{:}T' \mid e'\})$. By Lemma 32, we finish. ☐

Lemma 43 (Constructor Inversion)**.** *If* $\Gamma \vdash C\langle e\rangle v : T$, *then*

- $\mathit{TypSpecOf}(C) = x{:}T_1 \rightarrowtail T_2 \rightarrowtail \tau\langle x\rangle$,
- $\Gamma \vdash v : T_2\,\{e/x\}$,
- $\Gamma \vdash \tau\langle e\rangle$, *and*
- $\tau\langle e\rangle \equiv \mathit{unref}(T)$.

Proof. Similarly to Lemma 40, by induction on the typing derivation. Only four rules can be applied to the constructor application.

Case (T_Ctr): Since $T = \tau\langle e\rangle$, we have $\tau\langle e\rangle \equiv \mathit{unref}(T)$ by Lemma 1 (reflexivity). By inversion, we finish.

Case (T_Conv): By inversion, we have $\varnothing \vdash C\langle e\rangle v : T'$ and $T' \equiv T$ for some T'. By the IH, we have $\mathit{TypSpecOf}(C) = x{:}T_1 \rightarrowtail T_2 \rightarrowtail \tau\langle x\rangle$ and $\varnothing \vdash v : T_2\,\{e/x\}$ and $\varnothing \vdash \tau\langle e\rangle$ and $\tau\langle e\rangle \equiv \mathit{unref}(T')$. Because $\mathit{unref}(T') \equiv \mathit{unref}(T)$ by Lemma 39, we have $\tau\langle e\rangle \equiv \mathit{unref}(T)$ by Lemma 1 (transitivity). By Lemma 32, we finish.

Case (T_Forget): By inversion, we have $\varnothing \vdash C\langle e\rangle v : \{y{:}T \mid e'\}$ for some y and e'. By the IH, we have $\mathit{TypSpecOf}(C) = x{:}T_1 \rightarrowtail T_2 \rightarrowtail \tau\langle x\rangle$ and $\varnothing \vdash v : T_2\,\{e/x\}$ and $\varnothing \vdash \tau\langle e\rangle$ and $\tau\langle e\rangle \equiv \mathit{unref}(\{y{:}T \mid e'\})$. Since $\mathit{unref}(\{y{:}T \mid e'\}) = \mathit{unref}(T)$, we have $\tau\langle e\rangle \equiv \mathit{unref}(T)$. By Lemma 32, we finish.

Case (T_EXACT): We are given $\Gamma \vdash C\langle e\rangle v : \{y{:}T' \mid e'\}$ for some y, T' and e'. By inversion, we have $\varnothing \vdash C\langle e\rangle v : T'$. By the IH, we have $\mathit{TypSpecOf}(C) = x{:}T_1 \rightarrowtail T_2 \rightarrowtail \tau\langle x\rangle$ and $\varnothing \vdash v : T_2\,\{e/x\}$ and $\varnothing \vdash \tau\langle e\rangle$ and $\tau\langle e\rangle \equiv \mathit{unref}(T')$. Since $\mathit{unref}(T') = \mathit{unref}(\{y{:}T' \mid e'\})$, we have $\tau\langle e\rangle \equiv \mathit{unref}(\{y{:}T' \mid e'\})$. By Lemma 32, we finish. □

Lemma 44 (Canonical Forms)**.** *Suppose that $\varnothing \vdash v : T$.*

(1) If $\mathit{unref}(T) = \mathsf{Bool}$, then $v = \mathsf{true}$ or false.

(2) If $\mathit{unref}(T) = x{:}T_1 \to T_2$, then

(a) $v = \mathtt{fix}\ f(x{:}T_1'){:}T_2' = e$ for some f, T_1', T_2' and e, or

(b) $v = \langle T_2' \Leftarrow T_1'\rangle^{\ell}$ for some T_2', T_1' and ℓ.

(3) If $\mathit{unref}(T) = x{:}T_1 \times T_2$, then $v = (v_1, v_2)$ for some v_1 and v_2.

(4) If $\mathit{unref}(T) = \tau\langle e\rangle$, then $v = C\langle e'\rangle v'$ for some C, e' and v'.

Proof. By induction on the typing derivation.

Case (T_CONST): We are given $\varnothing \vdash c : \mathsf{Bool}$. By inversion, $c \in \{\mathsf{true}, \mathsf{false}\}$. Since $\mathit{unref}(\mathsf{Bool}) = \mathsf{Bool}$, we are in the case (1).

Case (T_VAR), (T_BLAME), (T_APP), (T_PROJi) for $i \in \{1,2\}$, (T_MATCH), (T_IF), (T_ACHECK), (T_WCHECK): Contradictory: v is a value.

Case (T_ABS): We are given $\varnothing \vdash \mathtt{fix}\ f(x{:}T_1){:}T_2 = e : x{:}T_1 \to T_2$. Since $\mathit{unref}(x{:}T_1 \to T_2) = x{:}T_1 \to T_2$, we are in the case (2).

Case (T_CAST): We are given $\varnothing \vdash \langle T_2 \Leftarrow T_1\rangle^{\ell} : T_1 \to T_2$. Since $\mathit{unref}(T_1 \to T_2) = T_1 \to T_2$, we are in the case (2).

Case (T_PAIR): We are given $\varnothing \vdash (v_1, v_2) : x{:}T_1 \times T_2$. Since $\mathit{unref}(x{:}T_1 \times T_2) = x{:}T_1 \times T_2$, we are in the case (3).

Case (T_CTR): We are given $\varnothing \vdash C\langle e'\rangle v' : \tau\langle e'\rangle$. Since $\mathit{unref}(\tau\langle e'\rangle) = \tau\langle e'\rangle$, we are in the case (4).

Case (T_CONV): By inversion, we have $\varnothing \vdash v : T'$ for some T' such that $T' \equiv T$. By Lemma 39, $\mathit{unref}(T') \equiv \mathit{unref}(T)$. By case analysis on $\mathit{unref}(T')$:

Case $\mathit{unref}(T') = \mathsf{Bool}$: By the IH, $v \in \{\mathsf{true}, \mathsf{false}\}$. By Lemma 34 (1), $\mathit{unref}(T) = \mathsf{Bool}$ and so we are in the case (1).

Case $\mathit{unref}(T') = x{:}T_1 \to T_2$: By the IH, v is a lambda abstraction or a cast. By Lemma 35 (1), $\mathit{unref}(T) = x{:}T_1' \to T_2'$ for some T_1' and T_2' and so we are in the case (2).

Case $\mathit{unref}(T') = x{:}T_1 \times T_2$: By the IH, $v = (v_1, v_2)$ for some v_1 and v_2. By Lemma 36 (1), $\mathit{unref}(T) = x{:}T_1' \times T_2'$ for some T_1' and T_2' and so we are in the case (3).

Case $\mathit{unref}(T') = \tau\langle e'\rangle$: By the IH, $v = C\langle e''\rangle v''$ for some e'' and v''. By Lemma 37 (1), $\mathit{unref}(T) = \tau\langle e'''\rangle$ for some e''' and so we are in the case (4).

Case (T_FORGET): By inversion, we have $\varnothing \vdash v : \{x{:}T \mid e\}$ for some x and e. Since $\mathit{unref}(T) = \mathit{unref}(\{x{:}T \mid e\})$, we finish by the IH.

Case (T_EXACT): We are given $\varnothing \vdash v : \{x{:}T' \mid e\}$ for some x, T' and e. By inversion, we have $\varnothing \vdash v : T'$. Since $\mathit{unref}(\{x{:}T' \mid e\}) = \mathit{unref}(T')$, we finish by the IH. □

Lemma 45 (Progress)**.** *If $\varnothing \vdash e : T$, then*

1. *$e \longrightarrow e'$ for some e',*
2. *e is a value, or*
3. *$e = \Uparrow\ell$ for some ℓ.*

Proof. By induction on the typing derivation.

Case (T_Const), (T_Blame), (T_Abs), (T_Cast), (T_Forget), (T_Exact): The term e is a blaming or a value.

Case (T_Var): Contradictory: $\varnothing \vdash x : T$ cannot be derived for any x.

Case (T_App): We are given $\varnothing \vdash e_1\, e_2 : T_2\,\{e_2/x\}$ for some e_1, e_2, T_2 and x. By inversion, we have $\varnothing \vdash e_1 : x{:}T_1 \to T_2$ and $\varnothing \vdash e_2 : T_1$ for some T_1.

By the IH, e_1 and e_2 are reducible, values, or blamings. If e_1 is reducible or a blaming, then $e_1\, e_2$ steps by one of evaluation rules. If e_1 is a value and e_2 is reducible or a blaming, then $e_1\, e_2$ steps by one of evaluation rules. Otherwise, if e_1 and e_2 are values, then there are two cases which we consider on e_1 by Lemma 44.

Case $e_1 = \mathtt{fix}\ f(x{:}T_1') = e_{12}$: The term $e_1\, e_2$ steps by (E_Red)/(R_Beta).

Case $e_1 = \langle T_1' \Leftarrow T_2' \rangle^{\ell}$: If T_2' is a refinement type, then we finish by (E_Red)/(R_Forget). In the following, we suppose that T_2' is not a refinement type. By Lemma 41, we have $T_1' \parallel T_2'$ and $T_2' \to T_1' \equiv x{:}T_1 \to T_2$ We perform case analysis on T_1'.

Case $T_1' = \mathsf{Bool}$: It is found from $\mathsf{Bool} \parallel T_2'$ that $T_2' = \mathsf{Bool}$ since T_2' is not a refinement type. We then finish by (E_Red)/(R_Base).

Case $T_1' = y{:}T_{11} \to T_{12}$: It is found that from $y{:}T_{11} \to T_{12} \parallel T_2'$ that $T_2' = y{:}T_{21} \to T_{22}$ for some T_{21} and T_{22} since T_2' is not a refinement type. We then finish by (E_Red)/(R_Fun).

Case $T_1' = y{:}T_{11} \times T_{12}$: It is found that from $y{:}T_{11} \times T_{12} \parallel T_2'$ that $T_2' = y{:}T_{21} \times T_{22}$ for some T_{21} and T_{22} since T_2' is not a refinement type. By Lemmas 35 and 36 (1), $T_1 = y{:}T_{11}' \times T_{12}'$ for some T_{11}' and T_{12}'. Since $\varnothing \vdash e_2 : T_1 = y{:}T_{11}' \times T_{12}'$ and e_2 is a value, we have $e_2 = (v_1, v_2)$ for some v_1 and v_2 by Lemma 44 (3). We then finish by (E_Red)/(R_Prod).

Case $T_1' = \tau_1\langle e_1' \rangle$: It is found that from $\Sigma \vdash \tau_1\langle e_1' \rangle \parallel T_2'$ that $T_2' = \tau_2\langle e_2' \rangle$ for some τ_2 and e_2' since T_2' is not a refinement type. If $\tau_1 = \tau_2$ and τ_1 is monomorphic, then we apply (E_Red)/(R_DatatypeMono); if $\tau_1 \neq \tau_2$ or τ_1 is not monomorphic, and $\delta(\langle \tau_1\langle e_1' \rangle \Leftarrow \tau_2\langle e_2' \rangle \rangle^{\ell}\, e_2)$ is defined, then (E_Red)/(R_Datatype); otherwise, (E_Red)/(R_DatatypeFail).

Case $T_1' = \{y{:}T_1'' \mid e_1''\}$: Since T_2' is not a refinement type, we finish by (E_Red)/(R_PreCheck).

Case (T_Pair): We are given $\varnothing \vdash (e_1, e_2) : x{:}T_1 \times T_2$ for some e_1, e_2, x, T_1 and T_2. By inversion, we have $\varnothing \vdash e_1 : T_1$ and $\varnothing \vdash e_2 : T_2\,\{e_1/x\}$. By the IH, e_1 and e_2 are reducible, values, or blamings. If e_1 is reducible or a blaming, then we finish by one of evaluation rules. If e_1 is a value and e_2 is reducible or a blaming, then we finish by one of evaluation rules. Otherwise, if e_1 and e_2 are values, then so is (e_1, e_2) is.

Case (T_Proj1): We are given $\varnothing \vdash e_1.1 : T_1$ for some e_1 and T_1. By inversion, we have $\varnothing \vdash e_1 : x{:}T_1 \times T_2$ for some x and T_2. By the IH, e_1 is reducible, a value, or a blaming. If e_1 is reducible or a blaming, then we finish by one of evaluation rules. Otherwise, if e_1 is a value, then $e_1 = (v_1, v_2)$ for some v_1 and v_2 by Lemma 44 (3), and so we finish by (E_Red)/(R_Proj1).

Case (T_Proj2): Similarly to the case for (T_Proj1). We are given $\varnothing \vdash e_2.2 : T_2\,\{e_2.1/x\}$ for some e_2, T_2, and x. By inversion, we have $\varnothing \vdash e_2 : x{:}T_1 \times T_2$ for some T_1. By the IH, e_2 is reducible, a value, or a blaming. If e_2 is reducible or a blaming, then we finish by one of evaluation rules. Otherwise, if e_2 is a value, then $e_2 = (v_1, v_2)$ for some v_1 and v_2 by Lemma 44 (3), and so we finish by (E_Red)/(R_Proj2).

Case (T_Ctr): We are given $\varnothing \vdash C\langle e_1 \rangle e_2 : \tau\langle e_1 \rangle$. By inversion, we have $\varnothing \vdash e_2 : T'\,\{e_1/x\}$ for some T' and x such that $\mathit{TypSpecOf}(C) = x{:}T'' \rightarrowtail T' \rightarrowtail \tau\langle x \rangle$. By the IH, e_2 is reducible, a value, or a blaming. If e_2 is reducible or a blaming, then we finish by one of evaluation rules. Otherwise, if e_2 is a value, then so is $C\langle e_1 \rangle e_2$.

Case (T_Match): We are given $\Gamma \vdash \mathsf{match}\, e_0\, \mathsf{with}\, \overline{C_i\, x_i \to e_i}^{\,i \in \{1,\ldots,n\}} : T$ for some e_0 and $\overline{C_i\, x_i \to e_i}^{\,i \in \{1,\ldots,n\}}$. By inversion, we have $\Gamma \vdash e_0 : \tau\langle e' \rangle$ for some τ and e'. By the IH, e_0 is reducible, a value, or a blaming. If e_0 is reducible or a blaming, then we finish by one of evaluation rules. Otherwise, if e_0 is a value, then, by Lemma 44 (4), we have $e_0 = C\langle e_1' \rangle v_2$ for some C, e_1' and v_2. By Lemmas 43 and 37, C is a constructor of τ. There therefore exists $j \in \{1, \ldots, n\}$ such that $C = C_j$ since patterns are exhaustive. By (R_Match), we finish.

Case (T_IF): We are given $\varnothing \vdash \mathsf{if}\, e_1 \,\mathsf{then}\, e_2 \,\mathsf{else}\, e_3 : T$ for some e_1, e_2 and e_3. By inversion, we have $\varnothing \vdash e_1 : \mathsf{Bool}$. By the IH, e_1 is reducible, a value, or a blaming. If e_1 is reducible or a blaming, then we finish by one of evaluation rules. Otherwise, if e_1 is a value, then e_1 is true or false by Lemma 44 (1). If e_1 is true (resp. false), then we finish by (R_IFTRUE) (resp. (R_IFFALSE)).

Case (T_WCHECK): We are given $\varnothing \vdash \langle\langle \{x{:}T' \,|\, e_1\}, e_2 \rangle\rangle^{\ell} : \{x{:}T' \,|\, e_1\}$ for some x, T', e_1, e_2 and ℓ. By inversion, we have $\varnothing \vdash e_2 : T'$. By the IH, e_2 is reducible, a value, or a blaming. If e_2 is reducible or a blaming, then we finish by one of evaluation rules. Otherwise, if e_2 is a value, we finish by (R_CHECK).

Case (T_ACHECK): We are given $\varnothing \vdash \langle \{x{:}T' \,|\, e_1\}, e_2, v \rangle^{\ell} : \{x{:}T' \,|\, e_1\}$ for some x, T', e_1, e_2, v and ℓ. By inversion, we have $\varnothing \vdash e_2 : \mathsf{Bool}$. By the IH, e_2 is reducible, a value, or a blaming. If e_2 is reducible or a blaming, then we finish by one of evaluation rules. Otherwise, if e_2 is a value, then e_2 is true or false by Lemma 44 (1). If e_2 is true (resp. false), then we finish by (R_OK) (resp. (R_FAIL)).

Case (T_CONV): By inversion, we have $\varnothing \vdash e : T'$. By the IH, we finish.

□

Lemma 46 (Context and Type Well-Formedness)**.**

1. *If* $\Gamma \vdash e : T$*, then* $\vdash \Gamma$ *and* $\Gamma \vdash T$*.*
2. *If* $\Gamma \vdash T$*, then* $\vdash \Gamma$*.*

Proof. By induction on the derivation of each judgment.

1. By case analysis on the typing derivation.

   Case (T_CONST): We are given $\Gamma \vdash c : T$ for some c. By inversion, we have $\vdash \Gamma$ and $T = \mathsf{Bool}$. By (WT_BASE), $\Gamma \vdash \mathsf{Bool}$.

   Case (T_VAR): We are given $\Gamma \vdash x : T$ for some x. By inversion, we have $\vdash \Gamma$ and $x{:}T \in \Gamma$. Let Γ_1 and Γ_2 be typing contexts such that $\Gamma_1, x{:}T, \Gamma_2 = \Gamma$. By inversion of $\vdash \Gamma$, we have $\Gamma_1 \vdash T$. Since for any $y{:}T' \in \Gamma_2$, $\Gamma_1, x{:}T, \Gamma_2' \vdash T'$ where $\Gamma_2 = \Gamma_2', y{:}T', \Gamma_2''$ for some Γ_2'', we have $\Gamma_1, x{:}T, \Gamma_2 \vdash T$ by Lemma 32.

   Case (T_BLAME): We are given $\Gamma \vdash \Uparrow\ell : T$ for some ℓ. By inversion, we have $\vdash \Gamma$ and $\varnothing \vdash T$. By Lemma 32, $\Gamma \vdash T$.

   Case (T_ABS): We are given $\Gamma \vdash \mathtt{fix}\ f(x{:}T_1){:}T_2 = e_2 : x{:}T_1 \to T_2$ for some f, x, T_1, T_2 and e_2. By inversion, we have $\Gamma, f{:}(x{:}T_1 \to T_2), x{:}T_1 \vdash e_2 : T_2$. By the IH, we have $\vdash \Gamma, f{:}(x{:}T_1 \to T_2), x{:}T_1$. By inversion of it, $\vdash \Gamma$ and $\Gamma \vdash x : T_1 \to T_2$.

   Case (T_CAST): We are given $\Gamma \vdash \langle T_1 \Leftarrow T_2 \rangle^{\ell} : x{:}T_2 \to T_1$ for some T_1, T_2, ℓ and x. Without loss of generality, we can suppose that x is fresh. By inversion, we have $\Gamma \vdash T_1$ and $\Gamma \vdash T_2$. By the IH, we have $\vdash \Gamma$. By Lemma 32, $\Gamma, x{:}T_2 \vdash T_1$. By (WT_FUN), we have $\Gamma \vdash x : T_2 \to T_1$.

   Case (T_APP): We are given $\Gamma \vdash e_1\, e_2 : T_2\, \{e_2/x\}$ for some T_2, e_2 and x. By inversion, we have $\Gamma \vdash e_1 : x{:}T_1 \to T_2$ and $\Gamma \vdash e_2 : T_1$. By the IH, we have $\vdash \Gamma$ and $\Gamma \vdash x : T_1 \to T_2$. By inversion of the latter, we have $\Gamma, x{:}T_1 \vdash T_2$. By Lemma 33, we have $\Gamma \vdash T_2\, \{e_2/x\}$.

   Case (T_PAIR): We are given $\Gamma \vdash (e_1, e_2) : x{:}T_1 \times T_2$ for some e_1, e_2, x, T_1 and T_2. By inversion, we have $\Gamma, x{:}T_1 \vdash T_2$. By the IH, $\vdash \Gamma, x{:}T_1$. By inversion of it, we have $\vdash \Gamma$. Since $\Gamma, x{:}T_1 \vdash T_2$, we finish by (WT_PROD).

   Case (T_PROJ1): We are given $\Gamma \vdash e'.1 : T$ for some e'. By inversion, we have $\Gamma \vdash e' : x{:}T \times T'$ for some x and T'. By the IH, we have $\vdash \Gamma$ and $\Gamma \vdash x{:}T \times T'$. By inversion of the latter, we have $\Gamma \vdash T$.

   Case (T_PROJ2): we are given $\Gamma \vdash e'.2 : T_2\, \{e'.1/x\}$ for some e', T_2 and x. By inversion, we have $\Gamma \vdash e' : x{:}T_1 \times T_2$ for some T_1. By the IH, $\vdash \Gamma$ and $\Gamma \vdash x{:}T_1 \times T_2$. By inversion of the latter, we have $\Gamma, x{:}T_1 \vdash T_2$. Since $\Gamma \vdash e' : x{:}T_1 \times T_2$, we have $\Gamma \vdash e'.1 : T_1$ by (T_PROJ1). By Lemma 33, we have $\Gamma \vdash T_2\, \{e'.1/x\}$.

   Case (T_CTR): We are given $\Gamma \vdash C\langle e_1 \rangle e_2 : \tau\langle e_1 \rangle$ for some C, e_1, e_2 and τ. By inversion, we have $\Gamma \vdash \tau\langle e_1 \rangle$. By the IH, we have $\vdash \Gamma$.

Case (T_Match): We are given $\Gamma \vdash \mathsf{match}\, e_0\, \mathsf{with}\, \overline{C_i\, x_i \rightarrow e_i}^{\,i} : T$ for some e_0 and $\overline{C_i\, x_i \rightarrow e_i}^{\,i}$. By inversion, we have $\Gamma \vdash T$. By the IH, we have $\vdash \Gamma$.

Case (T_If): We are given $\Gamma \vdash \mathsf{if}\, e_1\, \mathsf{then}\, e_2\, \mathsf{else}\, e_3 : T$ for some e_1, e_2 and e_3. By inversion, we have $\Gamma \vdash e_2 : T$. By the IH, we have $\vdash \Gamma$ and $\Gamma \vdash T$.

Case (T_WCheck): We are given $\Gamma \vdash \langle\!\langle \{x{:}T_1 \,|\, e_1\}, e_2 \rangle\!\rangle^{\ell} : \{x{:}T_1 \,|\, e_1\}$ for some x, T_1, e_1, e_2 and ℓ. By inversion, we have $\vdash \Gamma$ and $\varnothing \vdash \{x{:}T_1 \,|\, e_1\}$. By Lemma 32, we have $\Gamma \vdash \{x{:}T_1 \,|\, e_1\}$.

Case (T_ACheck): We are given $\Gamma \vdash \langle \{x{:}T_1 \,|\, e_1\}, e_2, v \rangle^{\ell} : \{x{:}T_1 \,|\, e_1\}$ for some x, T_1, e_1, e_2, v and ℓ. By inversion, we have $\vdash \Gamma$ and $\varnothing \vdash \{x{:}T_1 \,|\, e_1\}$. By Lemma 32, we have $\Gamma \vdash \{x{:}T_1 \,|\, e_1\}$.

Case (T_Conv): By inversion, we have $\vdash \Gamma$ and $\varnothing \vdash T$. By Lemma 32, we have $\Gamma \vdash T$.

Case (T_Forget): We are given $\Gamma \vdash v : T$ for some v. By inversion, we have $\vdash \Gamma$ and $\varnothing \vdash v : \{x{:}T \,|\, e'\}$ for some x and e'. By the IH, $\varnothing \vdash \{x{:}T \,|\, e'\}$. By inversion of it, we have $\varnothing \vdash T$. By Lemma 32, $\Gamma \vdash T$.

Case (T_Exact): We are given $\Gamma \vdash v : \{x{:}T' \,|\, e'\}$ for some v, x, T' and e'. By inversion, we have $\vdash \Gamma$ and $\varnothing \vdash \{x{:}T' \,|\, e'\}$. By Lemma 32, we finish.

2. By case analysis on the well-formedness derivation.

Case (WT_Base): We are given $\Gamma \vdash \mathsf{Bool}$ for some Bool. By inversion, we have $\vdash \Gamma$.

Case (WT_Fun): We are given $\Gamma \vdash x : T_1 \rightarrow T_2$ for some x, T_1 and T_2. By inversion, we have $\Gamma \vdash T_1$. By the IH, $\vdash \Gamma$.

Case (WT_Refine): We are given $\Gamma \vdash \{x{:}T' \,|\, e'\}$ for some x, T' and e'. By inversion, we have $\Gamma \vdash T'$. By the IH, $\vdash \Gamma$.

Case (WT_Prod): We are given $\Gamma \vdash x{:}T_1 \times T_2$ for some x, T_1 and T_2. By inversion, we have $\Gamma \vdash T_1$. By the IH, $\vdash \Gamma$.

Case (WT_Datatype): We are given $\Gamma \vdash \tau\langle e \rangle$ for some τ and e. By inversion and the IH, we finish. □

Lemma 47. *If $T_1 \parallel \{x{:}T_2 \,|\, e_2\}$, then $T_1 \parallel T_2$.*

Proof. By induction on $T_1 \parallel \{x{:}T_2 \,|\, e_2\}$. There are only two cases where $T_1 \parallel \{x{:}T_2 \,|\, e_2\}$ can be derived.

Case $\{x{:}T_1' \,|\, e_1'\} \parallel \{x{:}T_2 \,|\, e_2\}$: By inversion, we have $T_1' \parallel T_2$. By (C_RefineL), $\{x{:}T_1' \,|\, e_1'\} \parallel T_2$.

Case (C_RefineL): We are given $\{y{:}T_1' \,|\, e_1'\} \parallel \{x{:}T_2 \,|\, e_2\}$. By inversion, we have $T_1' \parallel \{x{:}T_2 \,|\, e_2\}$. By the IH, we have $T_1' \parallel T_2$. By (C_RefineL), we finish. □

Lemma 48. *If $T_1 \parallel T_2$, then $T_1 \parallel T_2\, \{e/x\}$ for any e and x.*

Proof. Straightforward by induction on $T_1 \parallel T_2$. □

Lemma 49 (Preservation)**.** *Suppose that $\varnothing \vdash e : T$.*

(1) If $e \rightsquigarrow e'$, then $\varnothing \vdash e' : T$.

(2) If $e \longrightarrow e'$, then $\varnothing \vdash e' : T$.

Proof.

1. By induction on the typing derivation.

Case (T_Const), (T_Var), (T_Blame), (T_Abs), (T_Cast), (T_Pair), (T_Ctr), (T_Forget) or (T_Exact): Trivial because e does not step in the reduction relation.

Case (T_App): We are given $\varnothing \vdash e_1\, e_2 : T_2\, \{e_2/x\}$ for some e_1, e_2, T_2 and x. Without loss of generality, we can suppose that x is fresh. By inversion, we have $\varnothing \vdash e_1 : x{:}T_1 \rightarrow T_2$ and $\varnothing \vdash e_2 : T_1$ for some T_1. By case analysis on the reduction rule applied.

Case (R_BETA): We are given $(\texttt{fix}\ f(x{:}T'_1){:}T'_2 = e_{12})\, v_2 \rightsquigarrow e_{12}\{v_2/x, \texttt{fix}\ f(x{:}T'_1) = e_{12}/f\}$ for some f, T'_1, T'_2, e_{12} and v_2. Without loss of generality, we can suppose that f is fresh. By Lemma 40, we have $f{:}(x{:}T'_1 \to T'_2), x{:}T'_1 \vdash e_{12} : T'_2$ and $f \notin \mathsf{FV}(T'_2)$ and $x{:}T'_1 \to T'_2 \equiv x{:}T_1 \to T_2$ for some T'_2. Note that x (resp. f) does not occur in T'_1 (resp. T'_1 and T'_2). By Lemma 46 and inversion, we have $\varnothing \vdash x{:}T'_1 \to T'_2$, and thus $\varnothing \vdash T'_1$. Because $\varnothing \vdash e_1 : x{:}T'_1 \to T'_2$ by Lemma 1 (symmetry) and (T_CONV), we have $x{:}T'_1 \vdash e_{12}\{e_1/f\} : T'_2$ by Lemma 33. Since $T_1 \equiv T'_1$ by Lemma 35, we have $\varnothing \vdash v_2 : T'_1$ by (T_CONV). By Lemma 33, $\varnothing \vdash e_{12}\{e_1/f, v_2/x\} : T'_2\{v_2/x\}$ (note that e_1 is closed). Since $T_2 \equiv T'_2$ by Lemma 35, we have $T_2\{v_2/x\} \equiv T'_2\{v_2/x\}$ by Lemma 4 (3). Because $\varnothing \vdash T_2\{v_2/x\}$ by Lemma 46, we have $\varnothing \vdash e_{12}\{e_1/f, v_2/x\} : T_2\{v_2/x\}$ by Lemma 1 (symmetry) and (T_CONV).

Case (R_BASE): We are given $\langle \mathsf{Bool} \Leftarrow \mathsf{Bool} \rangle^{\ell}\, v_2 \rightsquigarrow v_2$ for ℓ and v_2. By Lemmas 41, 35 and 34, we have $T_1 = T_2 = \mathsf{Bool}$. Since $T_2\{e_2/x\} = \mathsf{Bool}$ and so $\varnothing \vdash v_2 : \mathsf{Bool}$, we finish.

Case (R_FUN): We are given

$$\langle y{:}T_{11} \to T_{12} \Leftarrow y{:}T_{21} \to T_{22} \rangle^{\ell}\, v_2 \rightsquigarrow \lambda y{:}T_{11}.(\lambda z{:}T_{21}.\langle T_{12} \Leftarrow T_{22}\{z/y\} \rangle^{\ell}\, (v_2\, z))\, (\langle T_{21} \Leftarrow T_{11} \rangle^{\ell}\, y)$$

for some y, T_{11}, T_{12}, T_{21}, T_{22}, ℓ, v_2 and z such that z is fresh. By Lemma 41, we have $\varnothing \vdash y{:}T_{11} \to T_{12}$, $\varnothing \vdash y{:}T_{21} \to T_{22}$, $y{:}T_{11} \to T_{12} \parallel y{:}T_{21} \to T_{22}$ and $x{:}(y{:}T_{21} \to T_{22}) \to (y{:}T_{11} \to T_{12}) \equiv x{:}T_1 \to T_2$. Note that x does not occur in $y{:}T_{11} \to T_{12}$. By inversion of derivations, $\varnothing \vdash T_{11}$, $\varnothing \vdash T_{21}$, $y{:}T_{11} \vdash T_{12}$, $y{:}T_{21} \vdash T_{22}$, $T_{11} \parallel T_{21}$, and $T_{12} \parallel T_{22}$.

Since $T_{21} \parallel T_{11}$ by symmetry of the compatibility relation, we have $\varnothing \vdash \langle T_{21} \Leftarrow T_{11} \rangle^{\ell} : T_{11} \to T_{21}$ by (T_CAST). Since $\varnothing \vdash T_{11}$, we have $y{:}T_{11} \vdash \langle T_{21} \Leftarrow T_{11} \rangle^{\ell} : T_{11} \to T_{21}$ by Lemma 32. Since $y{:}T_{11} \vdash y : T_{11}$ by (T_VAR), we have $y{:}T_{11} \vdash \langle T_{21} \Leftarrow T_{11} \rangle^{\ell}\, y : T_{21}$ by (T_APP).

By Lemma 35, $y{:}T_{21} \to T_{22} \equiv T_1$ and $y{:}T_{11} \to T_{12} \equiv T_2$, and thus, by Lemma 35 (1), $T_1 = y{:}T'_{21} \to T'_{22}$ and $T_2 = y{:}T'_{11} \to T'_{12}$ for some T'_{21}, T'_{22}, T'_{11} and T'_{12}. Since $\varnothing \vdash v_2 : y{:}T'_{21} \to T'_{22}$ and $\varnothing \vdash y{:}T_{21} \to T_{22}$, we have $\varnothing \vdash v_2 : y{:}T_{21} \to T_{22}$ by Lemma 1 (symmetry) and (T_CONV). We have $z{:}T_{21} \vdash v_2 : y{:}T_{21} \to T_{22}$ by Lemma 32, and thus $z{:}T_{21} \vdash v_2\, z : T_{22}\{z/y\}$ by (T_VAR) and (T_APP).

Since $y{:}T_{21} \vdash T_{22}$, we have $z{:}T_{21}, y{:}T_{21} \vdash T_{22}$ and thus $y{:}T_{11}, z{:}T_{21} \vdash T_{22}\{z/y\}$ by Lemmas 33 and 32. Since $y{:}T_{11}, z{:}T_{21} \vdash T_{12}$ by Lemma 32, and $T_{12} \parallel T_{22}\{z/y\}$ by Lemma 48, we have $y{:}T_{11}, z{:}T_{21} \vdash \langle T_{12} \Leftarrow T_{22}\{z/y\} \rangle^{\ell} : T_{22}\{z/y\} \to T_{12}$ by (T_CAST).

By Lemma 32 and (T_APP), $y{:}T_{11}, z{:}T_{21} \vdash \langle T_{12} \Leftarrow T_{22}\{z/y\} \rangle^{\ell}\, (v_2\, z) : T_{12}$. By Lemma 32 and (T_ABS), we have $y{:}T_{11} \vdash \lambda z{:}T_{21}.\langle T_{12} \Leftarrow T_{22}\{z/y\} \rangle^{\ell}\, (v_2\, z) : T_{21} \to T_{12}$. (Note that z does not occur T_{12}.) Since $y{:}T_{11} \vdash \langle T_{21} \Leftarrow T_{11} \rangle^{\ell}\, y : T_{21}$, by (T_APP) we have $y{:}T_{11} \vdash (\lambda z{:}T_{21}.\langle T_{12} \Leftarrow T_{22}\{z/y\} \rangle^{\ell}\, (v_2\, z))\, (\langle T_{21} \Leftarrow T_{11} \rangle^{\ell}\, y) : T_{12}$. By Lemma 32 and (T_ABS), $\varnothing \vdash \lambda y{:}T_{11}.(\lambda z{:}T_{21}.\langle T_{12} \Leftarrow T_{22}\{z/y\} \rangle^{\ell}\, (v_2\, z))\, (\langle T_{21} \Leftarrow T_{11} \rangle^{\ell}\, y) : (y{:}T_{11} \to T_{12})$.

Since $y{:}T_{11} \to T_{12} \equiv T_2$, we have $(y{:}T_{11} \to T_{12})\{v_2/x\} \equiv T_2\{v_2/x\}$ by Lemma 4 (3). Since $(y{:}T_{11} \to T_{12})\{v_2/x\} = y{:}T_{11} \to T_{12}$ and $\varnothing \vdash T_2\{v_2/x\}$ by Lemma 46, we finish by (T_CONV).

Case (R_PROD): Similarly to the case for (R_FUN). We are given

$$\langle y{:}T_{11} \times T_{12} \Leftarrow y{:}T_{21} \times T_{22} \rangle^{\ell}\, (v_1, v_2) \rightsquigarrow (\lambda y{:}T_{11}.(y, \langle T_{12} \Leftarrow T_{22}\{v_1/y\} \rangle^{\ell}\, v_2))\, (\langle T_{11} \Leftarrow T_{21} \rangle^{\ell}\, v_1)$$

for some y, T_{11}, T_{12}, T_{21}, T_{22}, ℓ, v_1 and v_2. Without loss of generality, we can suppose that y is fresh. By Lemma 41, we have $\varnothing \vdash y{:}T_{11} \times T_{12}$ and $\varnothing \vdash y{:}T_{21} \times T_{22}$ and $y{:}T_{11} \times T_{12} \parallel y{:}T_{21} \times T_{22}$ and $x{:}(y{:}T_{21} \times T_{22}) \to y{:}T_{11} \times T_{12} \equiv x{:}T_1 \to T_2$. Note that x does not occur in $y{:}T_{11} \times T_{12}$. By inversion of derivations, $\varnothing \vdash T_{11}$ and $\varnothing \vdash T_{21}$ and $y{:}T_{11} \vdash T_{12}$ and $y{:}T_{21} \vdash T_{22}$ and $T_{11} \parallel T_{21}$ and $T_{12} \parallel T_{22}$.

By Lemma 42, we have $\varnothing \vdash v_1 : T'_{21}$ and $\varnothing \vdash v_2 : T'_{22}\{v_1/y\}$ and $y{:}T'_{21} \vdash T'_{22}$ and $y{:}T'_{21} \times T'_{22} \equiv \mathit{unref}(T_1)$ for some T'_{21} and T'_{22}. Since $y{:}T_{21} \times T_{22} \equiv T_1$ by Lemma 35, we have $y{:}T_{21} \times T_{22} \equiv y{:}T'_{21} \times T'_{22}$, and thus $T_{21} \equiv T'_{21}$ and $T_{22} \equiv T'_{22}$ by Lemma 36. Since $\varnothing \vdash T_{21}$, we have $\varnothing \vdash v_1 : T_{21}$ by Lemma 1 (symmetry) and (T_CONV). Therefore, we have $\varnothing \vdash \langle T_{11} \Leftarrow T_{21} \rangle^{\ell}\, v_1 : T_{11}$ by (T_CAST) and (T_APP).

Since $y{:}T_{21} \vdash T_{22}$ and $\varnothing \vdash v_1 : T_{21}$, we have $y{:}T_{11} \vdash T_{22}\{v_1/y\}$ by Lemmas 33 and 32. By Lemma 48, $T_{12} \parallel T_{22}\{v_1/y\}$. By (T_CAST), we have $y{:}T_{11} \vdash \langle T_{12} \Leftarrow T_{22}\{v_1/y\} \rangle^{\ell} : T_{22}\{v_1/y\} \to T_{12}$. Since $T_{22} \equiv T'_{22}$, we have $T_{22}\{v_1/y\} \equiv T'_{22}\{v_1/y\}$ by Lemma 4 (3). Since $\varnothing \vdash T_{22}\{v_1/y\}$ by Lemma 33, we have $\varnothing \vdash v_2 : T_{22}\{v_1/y\}$ by Lemma 1 (symmetry) and (T_CONV). By Lemma 32 and (T_APP), $y{:}T_{11} \vdash \langle T_{12} \Leftarrow T_{22}\{v_1/y\} \rangle^{\ell}\, v_2 : T_{12}$.

Let z be a fresh variable. Since $z{:}T_{11}, y{:}T_{11} \vdash \langle T_{12} \Leftarrow T_{22}\{v_1/y\} \rangle^{\ell}\, v_2 : T_{12}$ by Lemma 32, we have $z{:}T_{11} \vdash (\langle T_{12} \Leftarrow T_{22}\{v_1/y\} \rangle^{\ell}\, v_2)\{z/y\} : T_{12}\{z/y\}$ by Lemma 33. Since $z{:}T_{11} \vdash z : T_{11}$ by (T_VAR),

and $z{:}T_{11}, y{:}T_{11} \vdash T_{12}$ by Lemmas 32 and 33, we have $z{:}T_{11} \vdash (z, (\langle T_{12} \Leftarrow T_{22}\,\{v_1/y\}\rangle^\ell\, v_2)\,\{z/y\}) :$ $y{:}T_{11} \times T_{12}$ by Lemma 32 and (T_Pair). By Lemmas 32 and 33,
$y{:}T_{11} \vdash (z, (\langle T_{12} \Leftarrow T_{22}\,\{v_1/y\}\rangle^\ell\, v_2)\,\{z/y\})\,\{y/z\} : (y{:}T_{11} \times T_{12})\,\{y/z\}$, that is,

$$y{:}T_{11} \vdash (y, (\langle T_{12} \Leftarrow T_{22}\,\{v_1/y\}\rangle^\ell\, v_2)) : (y{:}T_{11} \times T_{12}).$$

By Lemma 32 and (T_Abs), $\varnothing \vdash \lambda y{:}T_{11}.(y, \langle T_{12} \Leftarrow T_{22}\,\{v_1/y\}\rangle^\ell\, v_2) : T_{11} \to y{:}T_{11} \times T_{12}$. By (T_App), $\varnothing \vdash (\lambda y{:}T_{11}.(y, \langle T_{12} \Leftarrow T_{22}\,\{v_1/y\}\rangle^\ell\, v_2))\,(\langle T_{11} \Leftarrow T_{21}\rangle^\ell\, v_1) : y{:}T_{11} \times T_{12}$.
Since $y{:}T_{11} \times T_{12} \equiv T_2$ by Lemma 36, we have $(y{:}T_{11} \times T_{12})\,\{v_2/x\} \equiv T_2\,\{v_2/x\}$ by Lemma 4 (3). Since $(y{:}T_{11} \times T_{12})\,\{v_2/x\} = y{:}T_{11} \times T_{12}$ and $\varnothing \vdash T_2\,\{v_2/x\}$ by Lemma 46, we finish by (T_Conv).

Case (R_Forget): We are given $\langle T_1' \Leftarrow \{y{:}T_2' \,|\, e_2'\}\rangle^\ell\, v_2 \leadsto \langle T_1' \Leftarrow T_2'\rangle^\ell\, v_2$ for some T_1', y, T_2', e_2' and v_2. Without loss of generality, we can suppose that y is fresh. By Lemma 41, we have $\varnothing \vdash T_1'$ and $\varnothing \vdash \{y{:}T_2' \,|\, e_2'\}$ and $T_1' \parallel \{y{:}T_2' \,|\, e_2'\}$ and $x{:}\{y{:}T_2' \,|\, e_2'\} \to T_1' \equiv x{:}T_1 \to T_2$. Note that x does not occur in T_1'. By inversion and Lemma 47, $\varnothing \vdash T_2'$ and $T_1' \parallel T_2'$.
By (T_Cast), we have $\varnothing \vdash \langle T_1' \Leftarrow T_2'\rangle^\ell : T_2' \to T_1'$. Since $\{y{:}T_2' \,|\, e_2'\} \equiv T_1$ by Lemma 35, we have $\varnothing \vdash v_2 : \{y{:}T_2' \,|\, e_2'\}$ by Lemma 1 (symmetry) and (T_Conv). By (T_Forget), $\varnothing \vdash v_2 : T_2'$. Thus, $\varnothing \vdash \langle T_1' \Leftarrow T_2'\rangle^\ell\, v_2 : T_1'$. Since $T_1' \equiv T_2$ by Lemma 35, $T_1'\,\{v_2/x\} \equiv T_2\,\{v_2/x\}$ by Lemma 4 (3). Since $T_1'\,\{v_2/x\} = T_1'$, we have $\varnothing \vdash \langle T_1' \Leftarrow T_2'\rangle^\ell\, v_2 : T_2\,\{v_2/x\}$ by Lemma 46 and (T_Conv).

Case (R_PreCheck): We are given $\langle \{y{:}T_1' \,|\, e_1'\} \Leftarrow T_2'\rangle^\ell\, v_2 \leadsto \langle\!\langle \{y{:}T_1' \,|\, e_1'\}, \langle T_1' \Leftarrow T_2'\rangle^\ell\, v_2\rangle\!\rangle^\ell$ for some y, T_1', e_1', T_2', ℓ and v_2. Without loss of generality, we can suppose that y is fresh. By Lemma 41, we have $\varnothing \vdash \{y{:}T_1' \,|\, e_1'\}$ and $\varnothing \vdash T_2'$ and $\{y{:}T_1' \,|\, e_1'\} \parallel T_2'$ and $x{:}T_2' \to \{y{:}T_1' \,|\, e_1'\} \equiv x{:}T_1 \to T_2$. Note that x does not occur in $\{y{:}T_1' \,|\, e_1'\}$. By inversion and Lemma 47, $\varnothing \vdash T_1'$ and $T_1' \parallel T_2'$.
By (T_Cast), we have $\varnothing \vdash \langle T_1' \Leftarrow T_2'\rangle^\ell : T_2' \to T_1'$. Since $T_2' \equiv T_1$ by Lemma 35, we have $\varnothing \vdash v_2 : T_2'$ by (T_Conv). Thus, by (T_App), $\varnothing \vdash \langle T_1' \Leftarrow T_2'\rangle^\ell\, v_2 : T_1'$. By (T_WCheck), $\varnothing \vdash \langle\!\langle \{y{:}T_1' \,|\, e_1'\}, \langle T_1' \Leftarrow T_2'\rangle^\ell\, v_2\rangle\!\rangle^\ell : \{y{:}T_1' \,|\, e_1'\}$. Since $\{y{:}T_1' \,|\, e_1'\} \equiv T_2$ by Lemma 35, we have $\{y{:}T_1' \,|\, e_1'\}\,\{v_2/x\} \equiv T_2\,\{v_2/x\}$. Since $\{y{:}T_1' \,|\, e_1'\}\,\{v_2/x\} = \{y{:}T_1' \,|\, e_1'\}$, we have $\varnothing \vdash \langle\!\langle \{y{:}T_1' \,|\, e_1'\}, \langle T_1' \Leftarrow T_2'\rangle^\ell\, v_2\rangle\!\rangle^\ell : T_2\,\{v_2/x\}$ by Lemma 46 and (T_Conv).

Case (R_Datatype): We are given

$$\langle \tau_1\langle e_1'\rangle \Leftarrow \tau_2\langle e_2'\rangle\rangle^\ell\, C_2\langle e'\rangle v \leadsto C_1\langle e_1'\rangle(\langle T_1''\,\{e_1'/y_1\} \Leftarrow T_2''\,\{e_2'/y_2\}\rangle^\ell\, v)$$

for some τ_1, e_1', τ_2, e_2', ℓ, C_2, e', v, C_1, T_1'', y_1, T_2'', and y_2 such that $\tau_1 \neq \tau_2$ or τ_1 is not monomorphic, and $C_1 = \delta(\langle \tau_1\langle e_1'\rangle \Leftarrow \tau_2\langle e_2'\rangle\rangle^\ell\, C_2\langle e'\rangle v)$ and, for $i \in \{1,2\}$, $\mathit{ArgTypeOf}(\tau_i) = y_i{:}T_i'$ and $\mathit{CtrArgOf}(C_i) = T_i''$.
Since the constructor choice function δ is well-formed, we find that $C_1 \in \mathit{CompatCtrsOf}(\tau_1, C_2)$, that is, $C_1 \in \mathit{CtrsOf}(\tau_1)$ and $T_1'' \parallel T_2''$ from well-formedness of the type definition environment. Also, $y_1{:}T_1' \vdash T_1''$ and $y_2{:}T_2' \vdash T_2''$ from well-formedness of the type definition environment.
By Lemma 48, $T_1''\,\{e_1'/y_1\} \parallel T_2''\,\{e_2'/y_2\}$. By Lemma 41, we have $\varnothing \vdash \tau_1\langle e_1'\rangle$ and $\varnothing \vdash \tau_2\langle e_2'\rangle$ and $x{:}\tau_2\langle e_2'\rangle \to \tau_1\langle e_1'\rangle \equiv x{:}T_1 \to T_2$. Note that x does not occur in $\tau_1\langle e_1'\rangle$. By inversion of derivations, and Lemma 33, we have $\varnothing \vdash T_1''\,\{e_1'/y_1\}$ and $\varnothing \vdash T_2''\,\{e_2'/y_2\}$. Thus by (T_Cast), $\varnothing \vdash \langle T_1''\,\{e_1'/y_1\} \Leftarrow T_2''\,\{e_2'/y_2\}\rangle^\ell : T_2''\,\{e_2'/y_2\} \to T_1''\,\{e_1'/y_1\}$.
By Lemma 43, $\varnothing \vdash v : T_2''\,\{e'/y_2\}$ and $\tau_2\langle e'\rangle \equiv \mathit{unref}(T_1)$. Since $\tau_2\langle e_2'\rangle \equiv \mathit{unref}(T_1)$ by Lemmas 35 and 39, we have $\tau_2\langle e'\rangle \equiv \tau_2\langle e_2'\rangle$ by Lemma 35 and Lemma 1 (transitivity). Thus, $e' \equiv e_2'$ by Lemma 37. Since $T_2''\,\{e'/y_2\} \equiv T_2''\,\{e_2'/y_2\}$ by Lemma 3 (3), we have $\varnothing \vdash v : T_2''\,\{e_2'/y_2\}$ by (T_Conv). By (T_App), we have $\varnothing \vdash \langle T_1''\,\{e_1'/y_1\} \Leftarrow T_2''\,\{e_2'/y_2\}\rangle^\ell\, v : T_1''\,\{e_1'/y_1\}$. By inversion of $\varnothing \vdash \tau_1\langle e_1'\rangle$, we have $\varnothing \vdash e_1' : T_1'$. Thus, by (T_Ctr), $\varnothing \vdash C_1\langle e_1'\rangle(\langle T_1''\,\{e_1'/y_1\} \Leftarrow T_2''\,\{e_2'/y_2\}\rangle^\ell\, v) :$ $\tau_1\langle e_1'\rangle$.
By Lemma 35, we have $\tau_1\langle e_1'\rangle \equiv T_2$. Since $\tau_1\langle e_1'\rangle\,\{C_2\langle e'\rangle v/x\} = \tau_1\langle e_1'\rangle$, we have $\tau_1\langle e_1\rangle \equiv T_2\,\{C_2\langle e'\rangle v/x\}$ by Lemma 4 (3). By Lemma 46 and (T_Conv), we finish.

Case (R_DatatypeMono): We are given $\langle \tau \Leftarrow \tau\rangle^\ell\, v_2 \leadsto v_2$ for some τ, ℓ and v_2. By Lemma 41, $x{:}\tau \to \tau \equiv x{:}T_1 \to T_2$. Note that x does not occur in τ. By Lemma 35, $\tau \equiv T_1$ and $\tau \equiv T_2$, and so $T_1 \equiv T_2$ by Lemma 1. Since $T_1\,\{v_2/x\} = T_1$ by Lemma 46, $T_1 \equiv T_2\,\{v_2/x\}$ by Lemma 4 (3). Since $\varnothing \vdash v_2 : T_1$, we have $\varnothing \vdash v_2 : T_2\,\{v_2/x\}$ by Lemma 46 and (T_Conv).

Case (R_DatatypeFail): We are given $\langle \tau_1\langle e_1'\rangle \Leftarrow \tau_2\langle e_2'\rangle\rangle^\ell\, v_2 \leadsto \Uparrow\ell$ for some τ_1, e_1', τ_2, e_2', ℓ and v_2. By Lemma 46 and (T_Blame), we finish.

Case (T_Proj1): We are given $\varnothing \vdash e_1.1 : T$ for some e_1. By inversion, we have $\varnothing \vdash e_1 : x{:}T \times T_2$ for some x and T_2. The term steps only by (R_Proj1): $(v_1, v_2).1 \rightsquigarrow v_1$ for some v_1 and v_2 such that $e_1 = (v_1, v_2)$. By Lemma 42, we have $\varnothing \vdash v_1 : T_1'$ and $x{:}T_1' \times T_2' \equiv x{:}T \times T_2$ for some T_1' and T_2'. By Lemma 36, we have $T_1' \equiv T$. Since $\varnothing \vdash T$ by Lemma 46, we have $\varnothing \vdash v_1 : T$ by (T_Conv).

Case (T_Proj2): We are given $\varnothing \vdash e_2.2 : T_2\,\{e_2.1/x\}$ for some e_2, T_2 and x. By inversion, we have $\varnothing \vdash e_2 : x{:}T_1 \times T_2$ for some T_1. The term steps only by (R_Proj2): $(v_1, v_2).2 \rightsquigarrow v_2$ for some v_1 and v_2 such that $e_2 = (v_1, v_2)$.

By Lemma 42, we have $\varnothing \vdash v_2 : T_2'\,\{v_1/x\}$ and $x{:}T_1' \times T_2' \equiv x{:}T_1 \times T_2$ for some T_1' and T_2'. Since $(v_1, v_2).1 \longrightarrow v_1$ by (E_Red)/(R_Proj1), we have $T_2'\,\{(v_1, v_2).1/x\} \equiv T_2'\,\{v_1/x\}$ by Lemmas 2 and 3 (3). Since $T_2' \equiv T_2$ by Lemma 36, we have $T_2'\,\{(v_1, v_2).1/x\} \equiv T_2\,\{(v_1, v_2).1/x\}$ by Lemma 4 (3), and thus $T_2'\,\{v_1/x\} \equiv T_2\,\{(v_1, v_2).1/x\}$ by Lemma 1 (symmetry and transitivity). Since $\varnothing \vdash T_2\,\{(v_1, v_2).1/x\}$ by Lemma 46, we have $\varnothing \vdash v_2 : T_2\,\{(v_1, v_2).1/x\}$ by (T_Conv).

Case (T_Match): We are given $\varnothing \vdash \mathsf{match}\, e_0 \,\mathsf{with}\, \overline{C_i\, x_i \to e_i}^{\,i \in \{1,\dots,n\}} : T$ for some e_0 and $\overline{C_i\, x_i \to e_i}^{\,i \in \{1,\dots,n\}}$. By inversion, we have $\varnothing \vdash e_0 : \tau\langle e'' \rangle$ and $\varnothing \vdash T$ and $\mathit{CtrsOf}(\tau) = \overline{C_i}^{\,i \in \{1,\dots,n\}}$ and $\mathit{ArgTypeOf}(\tau) = y{:}T'$ and, for $i \in \{1, \dots, n\}$, $\mathit{CtrArgOf}(C_i) = T_i$ and $x_i{:}T_i\,\{e''/y\} \vdash e_i : T$. The term steps only by (R_Match):

$$\mathsf{match}\, C_j\langle e''' \rangle v' \,\mathsf{with}\, \overline{C_i\, x_i \to e_i}^{\,i \in \{1,\dots,n\}} \rightsquigarrow e_j\,\{v'/x_j\}$$

for some $j \in \{1, \dots, n\}$, e''', v' such that $e_0 = C_j\langle e''' \rangle v'$.

By Lemma 43, we have $\varnothing \vdash v' : T_j\,\{e'''/y\}$ and $\tau\langle e''' \rangle \equiv \tau\langle e'' \rangle$. Since $e''' \equiv e''$ by Lemma 37, we have $T_j\,\{e'''/y\} \equiv T_j\,\{e''/y\}$ by Lemma 3 (3). Since $x_j{:}T_j\,\{e''/y\} \vdash e_j : T$, we have $\varnothing \vdash T_j\,\{e''/y\}$ by Lemma 46 and inversion. Thus we have $\varnothing \vdash v' : T_j\,\{e''/y\}$ by (T_Conv). Since x_j does not occur in T, we have $\varnothing \vdash e_j\,\{v'/x_j\} : T$ by Lemma 33.

Case (T_If): We are given $\varnothing \vdash \mathsf{if}\, e_1 \,\mathsf{then}\, e_2 \,\mathsf{else}\, e_3 : T$ for some e_1, e_2 and e_3. By inversion, we have $\varnothing \vdash e_2 : T$ and $\varnothing \vdash e_3 : T$. Only two reduction rules can be applied to the term: (R_IfTrue) and (R_IfFalse). The case of (R_IfTrue) follows from $\varnothing \vdash e_2 : T$, and (R_IfFalse) from $\varnothing \vdash e_3 : T$.

Case (T_WCheck): We are given $\varnothing \vdash \langle\!\langle \{x{:}T_1 \,|\, e_1\}, e_2 \rangle\!\rangle^{\ell} : \{x{:}T_1 \,|\, e_1\}$ for some x, T_1, e_1, e_2 and ℓ. By inversion, we have $\varnothing \vdash \{x{:}T_1 \,|\, e_1\}$ and $\varnothing \vdash e_2 : T_1$. The term steps only by (R_Check): $\langle\!\langle \{x{:}T_1 \,|\, e_1\}, v_2 \rangle\!\rangle^{\ell} \rightsquigarrow \langle \{x{:}T_1 \,|\, e_1\}, e_1\,\{v_2/x\}, v_2 \rangle^{\ell}$ for some v_2 such that $e_2 = v_2$.

From $\varnothing \vdash \{x{:}T_1 \,|\, e_1\}$, we find that $x{:}T_1 \vdash e_1 : \mathsf{Bool}$. By Lemma 33, $\varnothing \vdash e_1\,\{v_2/x\} : \mathsf{Bool}$. Because $e_1\,\{v_2/x\} \longrightarrow^* e_1\,\{v_2/x\}$, we finish.

Case (T_ACheck): We are given $\varnothing \vdash \langle \{x{:}T_1 \,|\, e_1\}, e_2, v \rangle^{\ell} : \{x{:}T_1 \,|\, e_1\}$ for some x, T_1, e_1, e_2 and v. By inversion, we have $\varnothing \vdash \{x{:}T_1 \,|\, e_1\}$ and $\varnothing \vdash v : T_1$ and $e_1\,\{v/x\} \longrightarrow^* e_2$. Only two reduction rules can be applied to the term: (R_OK) and (R_Fail). The case of (R_OK) follows from (T_Exact), and (R_Fail) from (T_Blame).

Case (T_Conv): By inversion, we have $\varnothing \vdash e : T'$ and $T' \equiv T$ and $\varnothing \vdash T$ for some T'. If e steps to e', then we have $\varnothing \vdash e' : T'$ by the IH. By (T_Conv), we finish.

2. By induction on the typing derivation. If $e \longrightarrow \Uparrow\ell$ by (E_Blame), then we finish by Lemma 46 and (T_Blame). In the following, we suppose that e steps by (E_Red).

Case (T_Const), (T_Var), (T_Blame), (T_Abs), (T_Cast), (T_Forget) or (T_Exact): Trivial because e does not step in the evaluation relation.

Case (T_App): We are given $\varnothing \vdash e_1\, e_2 : T_2\,\{e_2/x\}$ for some e_1, e_2, T_2 and x. By inversion, we have $\varnothing \vdash e_1 : x{:}T_1 \to T_2$ and $\varnothing \vdash e_2 : T_1$ for some T_1.

If e_1 is not a value, then $e_1 \longrightarrow e_1'$ for some e_1' (noting e_1 is not a blaming; if so, (E_Blame) is applied to $e_1\, e_2$, but it is contradictory). By the IH, $\varnothing \vdash e_1' : x{:}T_1 \to T_2$ and thus $\varnothing \vdash e_1'\, e_2 : T_2\,\{e_2/x\}$ by (T_App).

If e_1 is a value but e_2 is not, then $e_2 \longrightarrow e_2'$ for some e_2'. By the IH, $\varnothing \vdash e_2' : T_1$ and thus $\varnothing \vdash e_1\, e_2' : T_2\,\{e_2'/x\}$ by (T_App). Because $T_2\,\{e_2'/x\} \equiv T_2\,\{e_2/x\}$ by Lemmas 2, 3 (3) and 1, we have $\varnothing \vdash e_1\, e_2' : T_2\,\{e_2/x\}$ by Lemma 46 and (T_Conv).

Otherwise, if e_1 and e_2 are values, then we finish by the case (1).

Case (T_PAIR): We are given $\varnothing \vdash (e_1, e_2) : x{:}T_1 \times T_2$ for some e_1, e_2, x, T_1 and T_2. By inversion, we have $\varnothing \vdash e_1 : T_1$ and $\varnothing \vdash e_2 : T_2\{e_1/x\}$ and $x{:}T_1 \vdash T_2$.

If e_1 is not a value, then $e_1 \longrightarrow e_1'$ for some e_1'. By the IH, $\varnothing \vdash e_1' : T_1$ and thus $\varnothing \vdash T_2\{e_1'/x\}$ by Lemma 33. Because $T_2\{e_1/x\} \equiv T_2\{e_1'/x\}$ by Lemmas 2 and 3 (3), we have $\varnothing \vdash e_2 : T_2\{e_1'/x\}$ by (T_CONV). Thus, by (T_PAIR), $\varnothing \vdash (e_1', e_2) : x{:}T_1 \times T_2$.

If e_1 is a value but e_2 is not, then $e_2 \longrightarrow e_2'$ for some e_2'. By the IH, $\varnothing \vdash e_2' : T_2\{e_1/x\}$ and thus $\varnothing \vdash (e_1, e_2') : x{:}T_1 \times T_2$.

Otherwise, if e_1 and e_2 are values, then so is (e_1, e_2).

Case (T_PROJ1): We are given $\varnothing \vdash e_1.1 : T$ for some e_1. By inversion, we have $\varnothing \vdash e_1 : x{:}T \times T_2$ for some x and T_2. If e_1 is not a value, then $e_1 \longrightarrow e_1'$ for some e_1'. By the IH, $\varnothing \vdash e_1' : x{:}T \times T_2$ and thus $\varnothing \vdash e_1'.1 : T$ by (T_PROJ1). Otherwise, if e_1 is a value, we finish by the case (1).

Case (T_PROJ2): We are given $\varnothing \vdash e_2.2 : T_2\{e_2.1/x\}$ for some e_2, T_2 and x. By inversion, we have $\varnothing \vdash e_2 : x{:}T_1 \times T_2$ for some T_1. If e_2 is not a value, then $e_2 \longrightarrow e_2'$ for some e_2'. By the IH, $\varnothing \vdash e_2' : x{:}T \times T_2$ and thus $\varnothing \vdash e_2'.2 : T_2\{e_2'.1/x\}$ by (T_PROJ2). Because $T_2\{e_2'.1/x\} \equiv T_2\{e_2.1/x\}$ by Lemmas 2, 3 (3) and 1, we have $\varnothing \vdash e_2'.2 : T_2\{e_2.1/x\}$ by Lemma 46 and (T_CONV). Otherwise, if e_2 is a value, we finish by the case (1).

Case (T_IF): We are given $\varnothing \vdash \mathsf{if}\, e_1\, \mathsf{then}\, e_2\, \mathsf{else}\, e_3 : T$ for some e_1, e_2 and e_3. By inversion, we have $\varnothing \vdash e_1 : \mathsf{Bool}$ and $\varnothing \vdash e_2 : T$ and $\varnothing \vdash e_3 : T$. If e_1 is not a value, $e_1 \longrightarrow e_1'$ for some e_1'. By the IH, $\varnothing \vdash e_1' : \mathsf{Bool}$ and thus $\varnothing \vdash \mathsf{if}\, e_1'\, \mathsf{then}\, e_2\, \mathsf{else}\, e_3 : T$ by (T_IF). Otherwise, if e_1 is a value, then we finish by the case (1).

Case (T_CTR): We are given $\varnothing \vdash C\langle e_1\rangle e_2 : \tau\langle e_1\rangle$ for some C, e_1, e_2 and τ. By inversion, we have $\mathit{TypSpecOf}(C) = x{:}T_1 \rightarrowtail T_2 \rightarrowtail \tau\langle x\rangle$ and $\varnothing \vdash e_1 : T_1$ and $\varnothing \vdash e_2 : T_2\{e_1/x\}$ and $\varnothing \vdash \tau\langle e_1\rangle$. If e_2 is not a value, then $e_2 \longrightarrow e_2'$ for some e_2'. By the IH, $\varnothing \vdash e_2' : T_2\{e_1/x\}$ and thus $\varnothing \vdash C\langle e_1\rangle e_2' : \tau\langle e_1\rangle$ by (T_CTR). Otherwise, if e_2 is a value, then so is $C\langle e_1\rangle e_2$.

Case (T_MATCH): We are given $\varnothing \vdash \mathsf{match}\, e_0\, \mathsf{with}\, \overline{C_i\, x_i \to e_i}^{\,i} : T$. By inversion, we have $\varnothing \vdash e_0 : \tau\langle e''\rangle$ and $\varnothing \vdash T$ and $\mathit{CtrsOf}(\tau) = \overline{C_i}^{\,i}$ and $\mathit{ArgTypeOf}(\tau) = y{:}T'$ and, for all i, $\mathit{CtrArgOf}(C_i) = T_i$ and $x_i{:}T_i\{e''/y\} \vdash e_i : T$. If e_0 is not a value, then $e_0 \longrightarrow e_0'$ for some e_0'. By the IH, $\varnothing \vdash e_0' : \tau\langle e''\rangle$ and thus $\varnothing \vdash \mathsf{match}\, e_0'\, \mathsf{with}\, \overline{C_i\, x_i \to e_i}^{\,i} : T$ by (T_MATCH). Otherwise, if e_0 is a value, then we finish by the case (1).

Case (T_WCHECK): We are given $\varnothing \vdash \langle\!\langle \{x{:}T_1 \mid e_1\}, e_2\rangle\!\rangle^{\ell} : \{x{:}T_1 \mid e_1\}$ for some x, T_1, e_1, e_2 and ℓ. By inversion, we have $\varnothing \vdash \{x{:}T_1 \mid e_1\}$ and $\varnothing \vdash e_2 : T_1$. If e_2 is not a value, then $e_2 \longrightarrow e_2'$ for some e_2'. By the IH, $\varnothing \vdash e_2' : T_1$ and thus $\varnothing \vdash \langle\!\langle \{x{:}T_1 \mid e_1\}, e_2'\rangle\!\rangle^{\ell} : \{x{:}T_1 \mid e_1\}$ by (T_WCHECK). Otherwise, if e_2 is a value, then we finish by the case (1).

Case (T_ACHECK): We are given $\varnothing \vdash \langle \{x{:}T_1 \mid e_1\}, e_2, v\rangle^{\ell} : \{x{:}T_1 \mid e_1\}$ for some x, T_1, e_1, e_2, v and ℓ. By inversion, we have $\varnothing \vdash \{x{:}T_1 \mid e_1\}$ and $\varnothing \vdash v : T_1$ and $\varnothing \vdash e_2 : \mathsf{Bool}$ and $e_1\{v/x\} \longrightarrow^* e_2$. If e_2 is not a value, then $e_2 \longrightarrow e_2'$ for some e_2'. By the IH, $\varnothing \vdash e_2' : \mathsf{Bool}$. Because $e_1\{v/x\} \longrightarrow^* e_2'$, we have $\varnothing \vdash \langle \{x{:}T_1 \mid e_1\}, e_2', v\rangle^{\ell} : \{x{:}T_1 \mid e_1\}$. Otherwise, if e_2 is a value, then we finish by the case (1).

Case (T_CONV): By inversion, we have $\varnothing \vdash e : T'$ and $T' \equiv T$ and $\varnothing \vdash T$ for some T'. Since $e \longrightarrow e'$, we have $\varnothing \vdash e' : T'$ by the IH. By (T_CONV), $\varnothing \vdash e' : T$. □

Definition 6. *We define a function* refines *from types to sets of pairs of a bound variable and a term, as follows.*

$$\begin{array}{lcl} \mathit{refines}(\{x{:}T \mid e\}) & = & \{(x, e)\} \cup \mathit{refines}(T) \\ \mathit{refines}(T) & = & \varnothing \qquad \textit{(If } T \textit{ is not a refinement type.)} \end{array}$$

In addition, we write $\vdash v : \mathit{refines}(T)$ *if (1)* v *is a closed value, and (2) for any* $(x, e) \in \mathit{refines}(T)$, $e\{v/x\} \longrightarrow^*$ true.

Lemma 50.

(1) If $T_1 \Rrightarrow T_2$*, then* $\vdash v : \mathit{refines}(T_1)$ *iff* $\vdash v : \mathit{refines}(T_2)$.

(2) If $T_1 \equiv T_2$*, then* $\vdash v : \mathit{refines}(T_1)$ *iff* $\vdash v : \mathit{refines}(T_2)$.

Proof.

1. From $T_1 \Rrightarrow T_2$, there exist some T, x, e'_1 and e'_2 such that $T_1 = T\{e'_1/x\}$ and $T_2 = T\{e'_2/x\}$ and $e'_1 \longrightarrow e'_2$. By induction on T.

   Case $T = \mathsf{Bool}$, $y{:}T'_1 \to T'_2$, $y{:}T'_1 \times T'_2$, or $\tau\langle e\rangle$: Obvious because $\mathit{refines}(T_1)$ and $\mathit{refines}(T_2)$ are empty.

   Case $T = \{y{:}T' \,|\, e'\}$: Without loss of generality, we suppose that y is a fresh variable. Since $T'\{e'_1/x\} \Rrightarrow T'\{e'_2/x\}$, it suffices to show that $e'\{e'_1/x\}\{v/y\} \longrightarrow^* \mathsf{true}$ iff $e'\{e'_2/x\}\{v/y\} \longrightarrow^* \mathsf{true}$ by the IH. For $i \in \{1,2\}$, since v and e'_i are closed values (recall that the evaluation relation is defined over closed terms), we have $e'\{e'_i/x\}\{v/y\} = e'\{v/y\}\{e'_i/x\}$. Since $e'\{v/y\}\{e'_1/x\} \Rrightarrow e'\{v/y\}\{e'_2/x\}$, we finish by Lemma 30.

2. By induction on $T_1 \equiv T_2$.

   Case $T_1 \Rrightarrow T_2$: By the case (1).

   Case transitivity and symmetry: By the IH(s).

□

Lemma 51. *If $\varnothing \vdash v : T$, then $\vdash v : \mathit{refines}(T)$.*

Proof. By induction on $\varnothing \vdash v : T$.

Case (T_Const), (T_Abs), (T_Cast), (T_Pair) or (T_Ctr): Obvious because $\mathit{refines}(T) = \{\}$.

Case (T_Var), (T_Blame), (T_App), (T_Proj1), (T_Proj2), (T_Match), (T_If), (T_WCheck) or (T_ACheck): Contradictory.

Case (T_Conv): By inversion, we have $\varnothing \vdash v : T'$ for some T' such that $T' \equiv T$. By the IH and Lemma 50 (2), we finish.

Case (T_Forget): By inversion, we have $\varnothing \vdash v : \{x{:}T \,|\, e\}$ for some x and e. By the IH, we finish.

Case (T_Exact): We are given $\varnothing \vdash v : \{x{:}T' \,|\, e'\}$ for some x, T' and e'. By inversion, we have $\varnothing \vdash v : T'$ and $e'\{v/x\} \longrightarrow^* \mathsf{true}$. Since $\mathit{refines}(\{x{:}T' \,|\, e'\}) = \mathit{refines}(T') \cup \{(x, e')\}$, we finish by the IH. □

Theorem 1 (Type Soundness)**.** *If $\varnothing \vdash e : T$, then*

1. *$e \longrightarrow^* v$ for some v such that $\varnothing \vdash v : T$ and $\vdash v : \mathit{refines}(T)$;*
2. *$e \longrightarrow^* \Uparrow\ell$ for some ℓ; or*
3. *there is an infinite sequence of evaluation $e \longrightarrow e_1 \longrightarrow \cdots$.*

Proof. Suppose that $e \longrightarrow^* e'$ for some e' such that e' cannot reduce. We show the theorem by mathematical induction on the number of evaluation steps of e.

1. 0: We know that e cannot reduce. Since $\varnothing \vdash e : T$, we find that e is a value or a blaming by Lemma 45. Moreover, if e is a value, then $\vdash e : \mathit{refines}(T)$ by Lemma 51.
2. $i+1$: We are given $e \longrightarrow e'' \longrightarrow^i e'$ for some e''. By Lemma 49 (2), $\varnothing \vdash e'' : T$ and thus we finish by the IH. □

Trans

input:

$\texttt{fix}\ f(y{:}T, x{:}\mathsf{int\ list}) = \mathsf{match}\ x\ \mathsf{with}\ [\,] \rightarrow e_1 \mid z_1 :: z_2 \rightarrow e_2$

returns:

let τ be a fresh type name **in**

let $\{T_i\}_i =$

$$\left\{ z_1{:}\mathsf{int} \times \{z_2{:}T_0 \mid e_0\} \;\middle|\; \begin{array}{l} (e_{\mathbf{opt}}, e) \in \mathit{GenContracts}(e_2), \\ (T_0, e_0) = \mathit{Aux}(\tau, e_{\mathbf{opt}}, e) \end{array} \right\} \textbf{ in}$$

let D and $\overline{D_i}^{\,i}$ be fresh constructor names, and
z be a fresh variable **in**

$\mathsf{type}\ \tau\ \langle y{:}T \rangle = D \parallel [\,] : \{z{:}\mathsf{unit} \mid e_1\} \mid \overline{D_i \parallel (::) : T_i}^{\,i}$

where

$\mathit{Aux}(\tau, e_{\mathbf{opt}}, e) =$

let $e' = e\,\{\texttt{fix}\ f(y{:}T, x{:}\mathsf{int\ list}) = .../f\}$ **in**

match $e_{\mathbf{opt}}$ **with**

$\mid \mathit{Some}\, e'' \rightarrow (\tau\langle e''\rangle, \mathsf{let}\ z_2 = \langle \mathsf{int\ list} \Leftarrow \tau\langle e''\rangle\rangle^{\ell}\ z_2\ \mathsf{in}\ e')$

$\mid \mathit{None} \rightarrow (\mathsf{int\ list}, e')$

Figure 3: Translation.

5 Translation

We assume two things through this section. First, type definition environments include $\mathsf{int\ list}$. Second, we make type definition environment as well as constructor choice function explicit sometimes; we write $\langle \Sigma, \delta \rangle; \Gamma \vdash e : T$, $\langle \Sigma, \delta \rangle; \Gamma \vdash T$, and $\langle \Sigma, \delta \rangle \vdash \Gamma$ to expose both in typing judgments and $\delta \vdash e_1 \longrightarrow e_2$ and $\delta \vdash e_1 \longrightarrow^* e_2$ to expose constructor choice functions in evaluation. We still assume that type definition environments and constructor choice functions are well formed.

5.1 Definition

We define a class of predicate functions which can be given to the translation.

Definition 7. *A recursive predicate function* $F = \texttt{fix}\ f(y{:}T, x{:}\mathsf{int\ list}) = \mathsf{match}\ x\ \mathsf{with}\ [\,] \rightarrow e_1 \mid z_1 :: z_2 \rightarrow e_2$ *is translatable under* Σ *if*

- $(\Sigma, \varnothing); \varnothing \vdash F : T \rightarrow \mathsf{int\ list} \rightarrow \mathsf{Bool}$,
- $(\Sigma, \varnothing); y{:}T \vdash e_1 : \mathsf{Bool}$*, and*
- $(\Sigma, \varnothing); f{:}T \rightarrow \mathsf{int\ list} \rightarrow \mathsf{Bool}, y{:}T, z_1{:}\mathsf{int}, z_2{:}\mathsf{int\ list} \vdash e_2 : \mathsf{Bool}$.

We omit Σ *if it is clear from the context or not important.*

The empty constructor choice function means that F does not contain run-time terms. We refer to metasymbols (f, y, x, e_1, etc.) included by definition of F as ones with subscript F. For example, y in F is written as y^F when we want to emphasize that it is from F.

The translation algorithm *Trans* is shown in Figure 3, where uses the auxiliary function *GenContracts* defined in Figure 4.

5.2 Static Correctness

We first show that the new datatype generated from a translatable function by the translation algorithm is well formed.

Lemma 52 (Type Definition Weakening)**.** *Let* ς *be a type definition.*

(1) If $\langle \Sigma, \delta \rangle; \Gamma \vdash e : T$*, then* $\langle \Sigma, \varsigma, \delta \rangle; \Gamma \vdash e : T$*.*

$$
\begin{array}{lcl}
\mathit{GenContracts}\,(\mathsf{true}) = \{(\mathit{None}, \mathsf{true})\} & & \mathit{GenContracts}\,(\mathsf{false}) = \varnothing \\
\mathit{GenContracts}\,(\mathsf{if}\ f\ e_1\ z_2\ \mathsf{then}\ e_2\ \mathsf{else}\ e_3) & = & \{(\mathit{Some}\, e_1, e_2)\} \cup \\
 & & \{(e_{\mathbf{opt}}, \mathsf{if}\ f\ e_1\ z_2\ \mathsf{then\ false\ else}\ e_3') \mid (e_{\mathbf{opt}}, e_3') \in \mathit{GenContracts}\,(e_3)\} \\
 & & \hfill (\text{if } \mathsf{FV}\,(e_1) \subseteq \{y, z_1\}) \\
\mathit{GenContracts}\,(\mathsf{if}\ e_1\ \mathsf{then}\ e_2\ \mathsf{else}\ e_3) & = & \{(e_{\mathbf{opt}}, \mathsf{if}\ e_1\ \mathsf{then}\ e_2'\ \mathsf{else\ false}) \mid (e_{\mathbf{opt}}, e_2') \in \mathit{GenContracts}\,(e_2)\} \cup \\
 & & \{(e_{\mathbf{opt}}, \mathsf{if}\ e_1\ \mathsf{then\ false\ else}\ e_3') \mid (e_{\mathbf{opt}}, e_3') \in \mathit{GenContracts}\,(e_3)\} \\
 & & \hfill (\text{if a term of the form } f\, e\, z_2 \text{ occurs in } e_2 \text{ or } e_3) \\
\mathit{GenContracts}\,(\mathsf{match}\ e_0\ \mathsf{with}\ \overline{C_i\, x_i \to e_i}^{\,i \in \{1,\ldots,n\}}) & = & \bigcup_{j \in \{1,\ldots,n\}} \{(e_{\mathbf{opt}}, \mathsf{match}\ e_0\ \mathsf{with}\ \overline{C_i\, x_i \to e_i'}^{\,i \in \{1,\ldots,n\}}) \mid \\
 & & \hfill (e_{\mathbf{opt}}, e_j') \in \mathit{GenContracts}\,(e_j) \wedge \forall i \neq j .\ e_i' = \mathsf{false}\} \\
 & & \hfill (\text{if a term of the form } f\, e\, z_2 \text{ occurs in some } e_i) \\
\mathit{GenContracts}\,(e) & = & \{(\mathit{None}, e)\} \hfill (\text{otherwise})
\end{array}
$$

Figure 4: Generation of base contracts and argument terms to a manifest datatype.

(2) If $\langle \Sigma, \delta \rangle; \Gamma \vdash T$*, then* $\langle \Sigma, \varsigma, \delta \rangle; \Gamma \vdash T$.

(3) If $\langle \Sigma, \delta \rangle \vdash \Gamma$*, then* $\langle \Sigma, \varsigma, \delta \rangle \vdash \Gamma$.

Proof. Straightforward by induction on each derivation. □

Definition 8 (Free Variables in Typing Contexts)**.** *We write* $\mathsf{FV}\,(\Gamma)$ *to denote the set of free variables in a typing context* Γ*. Formally, it is defined as follows:*

$$
\begin{array}{lcl}
\mathsf{FV}\,(\varnothing) & = & \varnothing \\
\mathsf{FV}\,(\Gamma, x{:}T) & = & \mathsf{FV}\,(\Gamma) \cup (\mathsf{FV}\,(T) \backslash \mathsf{dom}\,(\Gamma))
\end{array}
$$

where $\mathsf{dom}\,(\Gamma)$ *means the set of binding variables in* Γ*.*

Lemma 53 (Strengthening)**.**

(1) If $\Gamma_1, x{:}T', \Gamma_2 \vdash e : T$ *and* $x \notin \mathsf{FV}\,(\Gamma_2) \cup \mathsf{FV}\,(e)$*, then* $\Gamma_1, \Gamma_2 \vdash e : T$.

(2) If $\Gamma_1, x{:}T', \Gamma_2 \vdash T$ *and* $x \notin \mathsf{FV}\,(\Gamma_2) \cup \mathsf{FV}\,(T)$*, then* $\Gamma_1, \Gamma_2 \vdash T$.

(3) If $\vdash \Gamma_1, x{:}T', \Gamma_2$ *and* $x \notin \mathsf{FV}\,(\Gamma_2)$*, then* $\vdash \Gamma_1, \Gamma_2$.

Proof. By induction on each derivation. The interesting cases are for (T_Abs), (T_App) and (T_Match).

1. By case analysis on the rule applied last.

   Case (T_Const): We are given $\Gamma_1, x{:}T', \Gamma_2 \vdash c : \mathsf{Bool}$. By inversion, we have $\vdash \Gamma_1, x{:}T', \Gamma_2$. By the IH, $\vdash \Gamma_1, \Gamma_2$ and thus $\Gamma_1, \Gamma_2 \vdash c : \mathsf{Bool}$ by (T_Const).

   Case (T_Var): We are given $\Gamma_1, x{:}T', \Gamma_2 \vdash y : T$. By inversion, we have $\vdash \Gamma_1, x{:}T', \Gamma_2$ and $y{:}T \in \Gamma_1, x{:}T', \Gamma_2$. By the IH, $\vdash \Gamma_1, \Gamma_2$. We find that $x \neq y$ from $x \notin \mathsf{FV}\,(y)$. Thus, $\Gamma_1, \Gamma_2 \vdash y : T$ by (T_Var).

   Case (T_Blame): We are given $\Gamma_1, x{:}T', \Gamma_2 \vdash \Uparrow\ell : T$. By inversion, we have $\vdash \Gamma_1, x{:}T', \Gamma_2$ and $\varnothing \vdash T$. By the IH, $\vdash \Gamma_1, \Gamma_2$ and thus $\Gamma_1, \Gamma_2 \vdash \Uparrow\ell : T$ by (T_Blame).

   Case (T_Abs): We are given $\Gamma_1, x{:}T', \Gamma_2 \vdash \mathtt{fix}\ f(y{:}T_1){:}T_2 = e_2 : y{:}T_1 \to T_2$. Without loss of generality, we can suppose that f and y are fresh for x. By inversion, we have $\Gamma_1, x{:}T', \Gamma_2, f{:}(y{:}T_1 \to T_2), y{:}T_1 \vdash e_2 : T_2$. Since $x \notin \mathsf{FV}\,(\Gamma_2) \cup \mathsf{FV}\,(\mathtt{fix}\ f(y{:}T_1){:}T_2 = e_2)$, we find that $x \notin \mathsf{FV}\,(\Gamma_2, f{:}(y{:}T_1 \to T_2), y{:}T_1) \cup \mathsf{FV}\,(e_2)$. Note that, thanks to type annotation T_2 in the lambda abstraction, we can find $x \notin \mathsf{FV}\,(T_2)$. Thus, by the IH, $\Gamma_1, \Gamma_2, f{:}(y{:}T_1 \to T_2), y{:}T_1 \vdash e_2 : T_2$. By (T_Abs), we finish.

   Case (T_Cast): We are given $\Gamma_1, x{:}T', \Gamma_2 \vdash \langle T_1 \Leftarrow T_2 \rangle^{\ell} : T_2 \to T_1$. By inversion, we have $\Gamma_1, x{:}T', \Gamma_2 \vdash T_1$ and $\Gamma_1, x{:}T', \Gamma_2 \vdash T_2$ and $T_1 \parallel T_2$. Since $x \notin \mathsf{FV}\,(\Gamma_2) \cup \mathsf{FV}\,(\langle T_1 \Leftarrow T_2 \rangle^{\ell})$, we find that $x \notin \mathsf{FV}\,(\Gamma_2) \cup \mathsf{FV}\,(T_1) \cup \mathsf{FV}\,(T_2)$. Thus, by the IHs, $\Gamma_1, \Gamma_2 \vdash T_1$ and $\Gamma_1, \Gamma_2 \vdash T_2$. By (T_Cast), we finish.

Case (T_APP): We are given $\Gamma_1, x{:}T', \Gamma_2 \vdash e_1\, e_2 : T_2\,\{e_2/y\}$. By inversion, we have $\Gamma_1, x{:}T', \Gamma_2 \vdash e_1 : y{:}T_1 \to T_2$ and $\Gamma_1, x{:}T', \Gamma_2 \vdash e_2 : T_1$. Since $x \notin \mathsf{FV}(\Gamma_2) \cup \mathsf{FV}(e_1\, e_2)$, we find that $x \notin \mathsf{FV}(\Gamma_2) \cup \mathsf{FV}(e_1) \cup \mathsf{FV}(e_2)$. Thus, by the IHs, $\Gamma_1, \Gamma_2 \vdash e_1 : y{:}T_1 \to T_2$ and $\Gamma_1, \Gamma_2 \vdash e_2 : T_1$. By (T_APP), we finish.

Case (T_PAIR): We are given $\Gamma_1, x{:}T', \Gamma_2 \vdash (e_1, e_2) : y{:}T_1 \times T_2$. Without loss of generality, we can suppose that y is fresh for x. By inversion, we have $\Gamma_1, x{:}T', \Gamma_2 \vdash e_1 : T_1$ and $\Gamma_1, x{:}T', \Gamma_2 \vdash e_2 : T_2\,\{e_1/y\}$ and $\Gamma_1, x{:}T', \Gamma_2, y{:}T_1 \vdash T_2$. Since $x \notin \mathsf{FV}(\Gamma_2) \cup \mathsf{FV}((e_1, e_2))$, we find that $x \notin \mathsf{FV}(\Gamma_2) \cup \mathsf{FV}(e_1) \cup \mathsf{FV}(e_2)$. Thus, by the IHs, $\Gamma_1, \Gamma_2 \vdash e_1 : T_1$ and $\Gamma_1, \Gamma_2 \vdash e_2 : T_2\,\{e_1/y\}$. By Lemma 46, $x \notin \mathsf{FV}(T_1) \cup \mathsf{FV}(T_2)$. Thus, by the IH, $\Gamma_1, \Gamma_2, y{:}T_1 \vdash T_2$. By (T_PAIR), we finish.

Case (T_PROJ1): We are given $\Gamma_1, x{:}T', \Gamma_2 \vdash e_1.1 : T$. By inversion, we have $\Gamma_1, x{:}T', \Gamma_2 \vdash e_1 : y{:}T_1 \times T_2$. Since $x \notin \mathsf{FV}(\Gamma_2) \cup \mathsf{FV}(e_1.1)$, we find that $x \notin \mathsf{FV}(\Gamma_2) \cup \mathsf{FV}(e_1)$. Thus, by the IH, $\Gamma_1, \Gamma_2 \vdash e_1 : y{:}T_1 \times T_2$. By (T_PROJ1), we finish.

Case (T_PROJ2): We are given $\Gamma_1, x{:}T', \Gamma_2 \vdash e_2.2 : T_2\,\{e_2.1/y\}$. By inversion, we have $\Gamma_1, x{:}T', \Gamma_2 \vdash e_2 : y{:}T_1 \times T_2$. Since $x \notin \mathsf{FV}(\Gamma_2) \cup \mathsf{FV}(e_2.2)$, we find that $x \notin \mathsf{FV}(\Gamma_2) \cup \mathsf{FV}(e_2)$. Thus, by the IH, $\Gamma_1, \Gamma_2 \vdash e_2 : y{:}T_1 \times T_2$. By (T_PROJ2), we finish.

Case (T_CTR): We are given $[G1, x : T', G2| - Ce1e2 : te1]$. By inversion, we have $\mathit{TypSpecOf}(C) = y{:}T_1 \rightarrowtail T_2 \rightarrowtail \tau\langle y\rangle$ and $\Gamma_1, x{:}T', \Gamma_2 \vdash e_1 : T_1$ and $\Gamma_1, x{:}T', \Gamma_2 \vdash e_2 : T_2\,\{e_1/y\}$ and $\Gamma_1, x{:}T', \Gamma_2 \vdash \tau\langle e_1\rangle$. Since $x \notin \mathsf{FV}(\Gamma_2) \cup \mathsf{FV}(C\langle e_1\rangle e_2)$, we find that $x \notin \mathsf{FV}(\Gamma_2) \cup \mathsf{FV}(e_1) \cup \mathsf{FV}(e_2)$. Thus, by the IHs, $\Gamma_1, \Gamma_2 \vdash e_1 : T_1$ and $\Gamma_1, \Gamma_2 \vdash T_2\,\{e_1/y\}$ and $\Gamma_1, \Gamma_2 \vdash \tau\langle e_1\rangle$. By (T_CTR), we finish.

Case (T_MATCH): We are given $\Gamma_1, x{:}T', \Gamma_2 \vdash \mathsf{match}\, e_0\, \mathsf{with}\, \overline{C_i\, y_i \to e_i}^{\,i} : T$. We can suppose that each y_i is fresh for x. By inversion, we have $\Gamma_1, x{:}T', \Gamma_2 \vdash e_0 : \tau\langle e'\rangle$ and $\Gamma_1, x{:}T', \Gamma_2 \vdash T$ and $\mathit{CtrsOf}(\tau) = \overline{C_i}^{\,i}$ and $\mathit{ArgTypeOf}(\tau) = y{:}T''$ and for any i, $\mathit{CtrArgOf}(C_i) = T_i$ and $\Gamma_1, x{:}T', \Gamma_2, y_i{:}T_i\,\{e'/y\} \vdash e_i : T$. Since $x \notin \mathsf{FV}(\Gamma_2) \cup \mathsf{FV}(\mathsf{match}\, e_0\, \mathsf{with}\, \overline{C_i\, y_i \to e_i}^{\,i})$, we find that $x \notin \mathsf{FV}(\Gamma_2) \cup \mathsf{FV}(e_0) \cup \bigcup_i \mathsf{FV}(e_i)$. Thus, by the IH, $\Gamma_1, \Gamma_2 \vdash e_0 : \tau\langle e'\rangle$. By Lemma 46 and its inversion, $x \notin \mathsf{FV}(e')$. From well-formedness of the type definition environment, $x \notin \mathsf{FV}(T_i)$. Thus, by the IHs, for any i, $\Gamma_1, \Gamma_2, y_i{:}T_i\,\{e'/y\} \vdash e_i : T$. By Lemma 46, $x \notin \mathsf{FV}(T)$ (noting τ has at least one constructor from well-formedness of the type definition environment). By the IH, $\Gamma_1, \Gamma_2 \vdash T$. By (T_MATCH), we finish.

Case (T_IF): We are given $\Gamma_1, x{:}T', \Gamma_2 \vdash \mathsf{if}\, e_1\, \mathsf{then}\, e_2\, \mathsf{else}\, e_3 : \mathsf{Bool}$. By inversion, we have $\Gamma_1, x{:}T', \Gamma_2 \vdash e_1 : \mathsf{Bool}$ and $\Gamma_1, x{:}T', \Gamma_2 \vdash e_2 : T$ and $\Gamma_1, x{:}T', \Gamma_2 \vdash e_3 : T$. Since $x \notin \mathsf{FV}(\Gamma_2) \cup \mathsf{FV}(\mathsf{if}\, e_1\, \mathsf{then}\, e_2\, \mathsf{else}\, e_3)$, we find that $x \notin \mathsf{FV}(\Gamma_2) \cup \mathsf{FV}(e_1) \cup \mathsf{FV}(e_2) \cup \mathsf{FV}(e_3)$. By the IHs, $\Gamma_1, \Gamma_2 \vdash e_1 : \mathsf{Bool}$ and $\Gamma_1, \Gamma_2 \vdash e_2 : T$ and $\Gamma_1, \Gamma_2 \vdash e_3 : T$. By (T_IF), we finish.

Case (T_ACHECK): We are given $\Gamma_1, x{:}T', \Gamma_2 \vdash \langle \{y{:}T_1 \mid e_1\}, e_2, v\rangle^{\ell} : \{y{:}T_1 \mid e_1\}$. By inversion, we have $\vdash \Gamma_1, x{:}T', \Gamma_2$ and $\varnothing \vdash \{y{:}T_1 \mid e_1\}$ and $\varnothing \vdash v : T_1$ and $\varnothing \vdash e_2 : \mathsf{Bool}$ and $e_1\,\{v/y\} \longrightarrow^* e_2$. By the IH, $\vdash \Gamma_1, \Gamma_2$. By (T_ACHECK), we finish.

Case (T_WCHECK): We are given $\Gamma_1, x{:}T', \Gamma_2 \vdash \langle\!\langle \{y{:}T_1 \mid e_1\}, e_2\rangle\!\rangle^{\ell} : \{y{:}T_1 \mid e_1\}$. By inversion, we have $\vdash \Gamma_1, x{:}T', \Gamma_2$ and $\varnothing \vdash \{y{:}T_1 \mid e_1\}$ and $\varnothing \vdash e_2 : T_1$. By the IH, $\vdash \Gamma_1, \Gamma_2$. By (T_ACHECK), we finish.

Case (T_CONV): By inversion, we have $\vdash \Gamma_1, x{:}T', \Gamma_2$ and $\varnothing \vdash e : T''$ and $T'' \equiv T$ and $\varnothing \vdash T$. By the IH, $\vdash \Gamma_1, \Gamma_2$. By (T_CONV), we finish.

Case (T_FORGET): We are given $\Gamma_1, x{:}T', \Gamma_2 \vdash v : T$. By inversion, we have $\vdash \Gamma_1, x{:}T', \Gamma_2$ and $\varnothing \vdash v : \{y{:}T \mid e'\}$. By the IH, $\vdash \Gamma_1, \Gamma_2$. By (T_FORGET), we finish.

Case (T_EXACT): We are given $\Gamma_1, x{:}T', \Gamma_2 \vdash v : \{y{:}T'' \mid e''\}$. By inversion, we have $\vdash \Gamma_1, x{:}T', \Gamma_2$ and $\varnothing \vdash v : T''$ and $\varnothing \vdash \{y{:}T'' \mid e''\}$ and $e''\,\{v/y\} \longrightarrow^* \mathsf{true}$. By the IH, $\vdash \Gamma_1, \Gamma_2$. By (T_EXACT), we finish.

2. By case analysis on the rule applied last.

Case (WT_BASE): We are given $\Gamma_1, x{:}T', \Gamma_2 \vdash \mathsf{Bool}$. By the IH and (WT_BASE), we finish.

Case (WT_FUN): We are given $\Gamma_1, x{:}T', \Gamma_2 \vdash y : T_1 \to T_2$. Without loss of generality, we can suppose that y is fresh for x. By inversion, we have $\Gamma_1, x{:}T', \Gamma_2 \vdash T_1$ and $\Gamma_1, x{:}T', \Gamma_2, y{:}T_1 \vdash T_2$. Since $x \notin \mathsf{FV}(\Gamma_2) \cup \mathsf{FV}(y{:}T_1 \to T_2)$, we find that $x \notin \mathsf{FV}(\Gamma_2) \cup \mathsf{FV}(T_1) \cup \mathsf{FV}(T_2)$. By the IHs, $\Gamma_1, \Gamma_2 \vdash T_1$ and $\Gamma_1, \Gamma_2, y{:}T_1 \vdash T_2$. By (WT_FUN), we finish.

Case (WT_PROD): We are given $\Gamma_1, x{:}T', \Gamma_2 \vdash y{:}T_1 \times T_2$. Without loss of generality, we can suppose that y is fresh for x. By inversion, we have $\Gamma_1, x{:}T', \Gamma_2 \vdash T_1$ and $\Gamma_1, x{:}T', \Gamma_2, y{:}T_1 \vdash T_2$. Since $x \notin \mathsf{FV}(\Gamma_2) \cup \mathsf{FV}(y{:}T_1 \times T_2)$, we find that $x \notin \mathsf{FV}(\Gamma_2) \cup \mathsf{FV}(T_1) \cup \mathsf{FV}(T_2)$. By the IHs, $\Gamma_1, \Gamma_2 \vdash T_1$ and $\Gamma_1, \Gamma_2, y{:}T_1 \vdash T_2$. By (WT_PROD), we finish.

Case (WT_REFINE): We are given $\Gamma_1, x{:}T', \Gamma_2 \vdash \{y{:}T'' \mid e''\}$. Without loss of generality, we can suppose that y is fresh for x. By inversion, we have $\Gamma_1, x{:}T', \Gamma_2 \vdash T''$ and $\Gamma_1, x{:}T', \Gamma_2, y{:}T'' \vdash e'' : \mathsf{Bool}$. Since $x \notin \mathsf{FV}(\Gamma_2) \cup \mathsf{FV}(\{y{:}T'' \mid e''\})$, we find that $x \notin \mathsf{FV}(\Gamma_2) \cup \mathsf{FV}(T'') \cup \mathsf{FV}(e'')$. Thus, by the IHs, $\Gamma_1, \Gamma_2 \vdash T''$ and $\Gamma_1, \Gamma_2, y{:}T'' \vdash e'' : \mathsf{Bool}$. By (WT_REFINE), we finish.

Case (WT_DATATYPE): We are given $\Gamma_1, x{:}T', \Gamma_2 \vdash \tau\langle e' \rangle$. By the IH and (WT_DATATYPE), we finish.

3. By case analysis on the rule applied last.

Case (WC_EMPTY): Obvious.

Case (WC_EXTENDVAR): If $\Gamma_2 = \emptyset$, then, by inversion, we have $\vdash \Gamma_1$ and thus we finish. Otherwise, if $\Gamma_2 = \Gamma_2', y{:}T''$, then, by inversion, $\vdash \Gamma_1, x{:}T', \Gamma_2'$ and $\Gamma_1, x{:}T', \Gamma_2' \vdash y : T''$. By the IHs and (WC_EXTENDVAR), we finish. ☐

Lemma 54 (Application Inversion)**.** *If $\Gamma \vdash e_1\, e_2 : T$, then*

- $\Gamma \vdash e_1 : x{:}T_1 \to T_2$,
- $\Gamma \vdash e_2 : T_1$, *and*
- $T_2\,\{e_2/x\} \equiv T$

for some x, T_1 and T_2.

Proof. Similarly to Lemma 40, by induction on the typing derivation. Only two rules can be applied to the application.

Case (T_APP): Since $T = T_2\,\{e_2/x\}$, we have $T_2\,\{e_2/x\} \equiv T$ by Lemma 1 (reflexivity). By inversion, we finish.

Case (T_CONV): By inversion, we have $\emptyset \vdash e_1\, e_2 : T'$ and $T' \equiv T$ for some T'. By the IH, we have $\emptyset \vdash e_1 : x{:}T_1 \to T_2$ and $\emptyset \vdash e_2 : T_1$ and $T_2\,\{e_2/x\} \equiv T'$. We have $T_2\,\{e_2/x\} \equiv T$ by Lemma 1 (transitivity). By Lemma 32, we finish. ☐

Lemma 55 (Variable Inversion)**.** *If $\Gamma \vdash x : T$, then $\vdash \Gamma$ and $x{:}T \in \Gamma$.*

Proof. Obvious because only (T_VAR) can drive $\Gamma \vdash x : T$. ☐

Lemma 56. *Let F be a translatable function, e be a subterm of ${e_2}^F$, $\Gamma_1 = f^F{:}T^F \to \mathsf{int\ list} \to \mathsf{Bool}, y^F{:}T^F, {z_1}^F{:}\mathsf{int}$, and Γ_2 be a typing context. If $\Gamma_1, \Gamma_2 \vdash e : \mathsf{Bool}$ and $(e_{\boldsymbol{opt}_0}, e_0) \in \mathit{GenContracts}(e)$, then:*

- *for any e', if $e_{\boldsymbol{opt}_0} = \mathit{Some}\, e'$, then $y^F{:}T^F, {z_1}^F{:}\mathsf{int} \vdash e' : T^F$; and*
- $\Gamma_1, \Gamma_2 \vdash e_0 : \mathsf{Bool}$.

Proof. By structural induction on e with case analysis on $\Gamma_1, \Gamma_2 \vdash e : \mathsf{Bool}$.

Case (T_CONST): Obvious because $\mathit{GenContracts}(\mathsf{true}) = \{(\mathit{None}, \mathsf{true})\}$ and $\mathit{GenContracts}(\mathsf{false}) = \emptyset$.

Case (T_VAR), (T_ABS), (T_CAST), (T_APP), (T_PAIR), (T_PROJi) for $i \in \{1, 2\}$, (T_CTR), (T_FORGET), (T_EXACT), (T_BLAME), (T_ACHECK), and (T_WCHECK): Obvious because $\mathit{GenContracts}(e) = \{(\mathit{None}, e)\}$.

Case (T_IF): We are given $\Gamma_1, \Gamma_2 \vdash \mathsf{if}\, e_1\, \mathsf{then}\, e_2\, \mathsf{else}\, e_3 : \mathsf{Bool}$. By inversion, we have $\Gamma_1, \Gamma_2 \vdash e_1 : \mathsf{Bool}$ and $\Gamma_1, \Gamma_2 \vdash e_2 : \mathsf{Bool}$ and $\Gamma_1, \Gamma_2 \vdash e_3 : \mathsf{Bool}$. There are three cases which we have to consider.

Case $e_1 = f^F\, e_1'\, {z_2}^F$ where $\mathsf{FV}(e_1') \subseteq \{y^F, {z_1}^F\}$: Then,

$$GenContracts(e) = \{(Some\, e_1', e_2)\} \cup \{(e_{\mathbf{opt}}, \mathsf{if}\ f^F\, e_1'\, {z_2}^F\ \mathsf{then\, false\, else}\ e_3') \mid (e_{\mathbf{opt}}, e_3') \in GenContracts(e_3)\}$$

We first show $y^F{:}T^F, {z_1}^F{:}\mathsf{int} \vdash e_1' : T^F$. Since $\Gamma_1, \Gamma_2 \vdash f^F\, e_1'\, {z_2}^F : \mathsf{Bool}$, we find that $\Gamma_1, \Gamma_2 \vdash f^F : x{:}T_1 \to T_2$ and $\Gamma_1, \Gamma_2 \vdash e_1' : T_1$ for some x, T_1 and T_2, by applying Lemma 54 twice. By Lemma 55, $x{:}T_1 \to T_2 = T^F \to \mathsf{int\ list} \to \mathsf{Bool}$ since $f^F{:}x{:}T_1 \to T_2 \in \Gamma_1$. Thus, $T_1 = T^F$ and so $\Gamma_1, \Gamma_2 \vdash e_1' : T^F$. Since $\mathsf{FV}(e_1') \subseteq \{y^F, {z_1}^F\}$, and $f^F \notin \mathsf{FV}(T^F)$ by Lemma 46, we have $y^F{:}T^F, {z_1}^F{:}\mathsf{int} \vdash e_1' : T^F$ by Lemma 53 (1). In addition, we have $\Gamma_1, \Gamma_2 \vdash e_2 : \mathsf{Bool}$ from the premise of the typing derivation.

Let $(e_{\mathbf{opt}}, e_3') \in GenContracts(e_3)$. It suffices to show that (1) for any e', if $e_{\mathbf{opt}} = Some\, e'$, then $y^F{:}T^F, {z_1}^F{:}\mathsf{int} \vdash e' : T^F$ and (2) $\Gamma_1, \Gamma_2 \vdash \mathsf{if}\ f^F\, e_1'\, {z_2}^F\ \mathsf{then\, false\, else}\ e_3' : \mathsf{Bool}$. The case (1) is shown by the IH. The case (2) is obvious by (T_If) because $\Gamma_1, \Gamma_2 \vdash \mathsf{false} : \mathsf{Bool}$ by Lemmas 46 and 32 and $\Gamma_1, \Gamma_2 \vdash e_3' : \mathsf{Bool}$ by the IH.

Case $e_1 \neq f^F\, e_1'\, {z_2}^F$ for any e_1' such that $\mathsf{FV}(e_1') \subseteq \{y^F, {z_1}^F\}$, and a term of the form $f^F\, e_1'\, {z_2}^F$ for some e_1' occurs in e_2 or e_3: Similarly to the above. We have

$$\begin{array}{rcl} GenContracts(e) & = & \{(e_{\mathbf{opt}}, \mathsf{if}\ e_1\ \mathsf{then}\ e_2'\ \mathsf{else\, false}) \mid (e_{\mathbf{opt}}, e_2') \in GenContracts(e_2)\} \cup \\ & & \{(e_{\mathbf{opt}}, \mathsf{if}\ e_1\ \mathsf{then\, false\, else}\ e_3') \mid (e_{\mathbf{opt}}, e_3') \in GenContracts(e_3)\}. \end{array}$$

Since $\Gamma_1, \Gamma_2 \vdash e_2 : \mathsf{Bool}$ and $\Gamma_1, \Gamma_2 \vdash e_3 : \mathsf{Bool}$, we finish by the IHs.

Case otherwise: Obvious because $GenContracts(e) = \{(None, e)\}$.

Case (T_Match): Similarly to the case for (T_If). We are given $\Gamma_1, \Gamma_2 \vdash \mathsf{match}\, e_0\, \mathsf{with}\, \overline{C_i\, x_i \to e_i}^{\,i \in \{1,\dots,n\}} : \mathsf{Bool}$. By inversion, we have $\Gamma_1, \Gamma_2 \vdash e_0 : \tau\langle e' \rangle$ and $ArgTypeOf(\tau) = x'{:}T'$ and, for any $i \in \{1, \dots, n\}$, $CtrArgOf(C_i) = T_i$ and $\Gamma_1, \Gamma_2, x_i{:}T_i\,\{e'/x'\} \vdash e_i : \mathsf{Bool}$ for some τ, e', x', T', and $\overline{T_i}^{\,i \in \{1,\dots,n\}}$.

If some e_i contains a term of the form $f^F\, e_1'\, {z_2}^F$ for some e_1', then we have

$$\begin{array}{rcl} GenContracts(e) & = & \bigcup_{j \in \{1,\dots,n\}} \{(e_{\mathbf{opt}}, \mathsf{match}\, e_0\, \mathsf{with}\, \overline{C_i\, x_i \to e_i''}^{\,i \in \{1,\dots,n\}}) \mid \\ & & \qquad (e_{\mathbf{opt}}, e_j'') \in GenContracts(e_j) \wedge \forall i \neq j\, .\, e_i'' = \mathsf{false}\}. \end{array}$$

We finish by the IHs with the fact that, for any i, $\Gamma_1, \Gamma_2, x_i{:}T_i\,\{e'/x'\} \vdash \mathsf{false} : \mathsf{Bool}$ by Lemmas 46 and 32, and so $\Gamma_1, \Gamma_2, x_i{:}T_i\,\{e'/x'\} \vdash e_i'' : \mathsf{Bool}$.

Otherwise, obvious because $GenContracts(e) = \{(None, e)\}$.

Case (T_Conv): By inversion, we have $\varnothing \vdash e : T$ and $T \equiv \mathsf{Bool}$. If $e = \mathsf{false}$, then obvious because $GenContracts(\mathsf{false}) = \varnothing$. Otherwise, since f^F (and ${z_2}^F$) does not occur in e, we have $GenContracts(e) = \{(None, e)\}$ (even if $e = \mathsf{true}$) and so we finish.

□

Lemma 57 (Translation Generates Well-Formed Datatype)**.** *Let Σ be a well-formed type definition environment and F be a translatable function under Σ. Then, $Trans(F)$ is well formed under Σ, that is, so is $\Sigma, Trans(F)$.*

Proof. By definition, $Trans(F) = \mathsf{type}\,\tau\,\langle y^F{:}T^F\rangle = D \parallel [\,] : \{z{:}\mathsf{unit} \mid {e_1}^F\} \mid \overline{D_i \parallel (::) : T_i}^{\,i}$ where z is fresh. It suffices to show that the type definition satisfies five conditions from definition of well-formedness of type definition under type definition environment.

(a) We show that τ has constructors more than zero, which is obvious.

(b) We show that $\Sigma; \varnothing \vdash T^F$. Since F is well typed, we have $\Sigma; \varnothing \vdash T^F$ by Lemma 46 and its inversion.

(c) We show that (1) $\Sigma, Trans(F); y^F{:}T^F \vdash \{z{:}\mathsf{unit} \mid {e_1}^F\}$ and (2) $\Sigma, Trans(F); y^F{:}T^F \vdash T_i$ for any i.

   (1) Since F is translatable under Σ, we have $(\Sigma, \varnothing); y^F{:}T^F \vdash {e_1}^F : \mathsf{Bool}$. By Lemma 52, $(\Sigma, Trans(F), \varnothing); y^F{:}T^F \vdash {e_1}^F : \mathsf{Bool}$. By Lemma 32 and (T_Refine), $(\Sigma, Trans(F), \varnothing); y^F{:}T^F \vdash \{z{:}\mathsf{unit} \mid {e_1}^F\}$.

(2) By definition of *GenContracts*, T_i is defined based on $\mathit{GenContracts}(e_2{}^F)$. Let $(e_{\mathbf{opt}}, e) \in \mathit{GenContracts}(e_2{}^F)$ and $\Gamma = f^F{:}T^F \rightarrow \mathsf{int\ list} \rightarrow \mathsf{Bool}, y^F{:}T^F, z_1{}^F{:}\mathsf{int}, z_2{}^F{:}\mathsf{int\ list}$. Since F is translatable under Σ, we have $(\Sigma, \varnothing); \Gamma \vdash e_2{}^F : \mathsf{Bool}$. By Lemma 56, $(\Sigma, \varnothing); \Gamma \vdash e : \mathsf{Bool}$. Since $(\Sigma, \varnothing); \varnothing \vdash F : T^F \rightarrow \mathsf{int\ list} \rightarrow \mathsf{Bool}$, we have $(\Sigma, \varnothing); y^F{:}T^F, z_1{}^F{:}\mathsf{int}, z_2{}^F{:}\mathsf{int\ list} \vdash e\,\{F/f^F\} : \mathsf{Bool}$ by Lemma 33. Note that T^F is closed by Lemma 46 and its inversion. By Lemma 52,

$$(\Sigma, \mathit{Trans}(F), \varnothing); y^F{:}T^F, z_1{}^F{:}\mathsf{int}, z_2{}^F{:}\mathsf{int\ list} \vdash e\,\{F/f^F\} : \mathsf{Bool}.$$

By case analysis on $e_{\mathbf{opt}}$, letting $\Gamma' = y^F{:}T^F, z_1{}^F{:}\mathsf{int}$.

Case $e_{\mathbf{opt}} = \mathit{Some}\, e''$: By Lemma 32 and (T_Abs),

$$(\Sigma, \mathit{Trans}(F), \varnothing); \Gamma' \vdash \lambda\, z_2{}^F{:}\mathsf{int\ list}.e\,\{F/f^F\} : \mathsf{int\ list} \rightarrow \mathsf{Bool}.$$

By Lemmas 56 and 52,

$$(\Sigma, \mathit{Trans}(F), \varnothing); \Gamma' \vdash e'' : T^F.$$

Thus,

$$(\Sigma, \mathit{Trans}(F), \varnothing); \Gamma' \vdash \tau\langle e'' \rangle$$

by (WT_Datatype). By (C_Datatype), $\Sigma, \mathit{Trans}(F) \vdash \tau\langle e'' \rangle \parallel \mathsf{int\ list}$. Since $(\Sigma, \mathit{Trans}(F), \varnothing); \Gamma' \vdash \mathsf{int\ list}$ by Lemmas 46 and 32 and (WT_Datatype), we find

$$(\Sigma, \mathit{Trans}(F), \varnothing); \Gamma' \vdash \langle \mathsf{int\ list} \Leftarrow \tau\langle e'' \rangle \rangle^{\ell} : \tau\langle e'' \rangle \rightarrow \mathsf{int\ list}$$

for any ℓ, by (T_Cast). By Lemma 32, (T_Var) and (T_App), we have

$$(\Sigma, \mathit{Trans}(F), \varnothing); \Gamma', z_2{}^F{:}\tau\langle e'' \rangle \vdash \langle \mathsf{int\ list} \Leftarrow \tau\langle e'' \rangle \rangle^{\ell}\, z_2{}^F : \mathsf{int\ list}.$$

Letting $e_0 = (\lambda\, z_2{}^F{:}\mathsf{int\ list}.e\,\{F/f^F\})\,(\langle \mathsf{int\ list} \Leftarrow \tau\langle e'' \rangle \rangle^{\ell}\, z_2{}^F)$, we have

$$(\Sigma, \mathit{Trans}(F), \varnothing); \Gamma', z_2{}^F{:}\tau\langle e'' \rangle \vdash e_0 : \mathsf{Bool}$$

by Lemma 32 and (T_App). Note that e_0 can be written as $\mathsf{let}\, z_2{}^F = \langle \mathsf{int\ list} \Leftarrow \tau\langle e'' \rangle \rangle^{\ell}\, z_2{}^F\ \mathsf{in}\, e\,\{F/f^F\}$. Letting $T_0 = \tau\langle e'' \rangle$, we have

$$(\Sigma, \mathit{Trans}(F), \varnothing); \Gamma' \vdash \{z_2{}^F{:}T_0 \mid e_0\}.$$

by (WT_Refine). Thus, by (WT_Prod),

$$(\Sigma, \mathit{Trans}(F), \varnothing); y^F{:}T^F \vdash z_1{}^F : \mathsf{int} \times \{z_2{}^F{:}T_0 \mid e_0\}.$$

Note that $T_i = z_1{}^F{:}\mathsf{int} \times \{z_2{}^F{:}T_0 \mid e_0\}$.

Case $e_{\mathbf{opt}} = \mathit{None}$: By (WT_Refine) and (WT_Prod), we have

$$(\Sigma, \mathit{Trans}(F), \varnothing); y^F{:}T^F \vdash z_1{}^F : \mathsf{int} \times \{z_2{}^F{:}\mathsf{int\ list} \mid e\,\{F/f^F\}\}.$$

Note that $T_i = z_1{}^F{:}\mathsf{int} \times \{z_2{}^F{:}\mathsf{int\ list} \mid e\,\{F/f^F\}\}$.

(d) We show that Σ includes $\mathsf{int\ list}$, which is proven by the assumption.

(e) We show that (1) $\Sigma, \mathit{Trans}(F) \vdash \{z{:}\mathsf{unit} \mid e_1{}^F\} \parallel \mathsf{unit}$ and (2) $\Sigma, \mathit{Trans}(F) \vdash T_i \parallel \mathsf{int} \times \mathsf{int\ list}$. The case (1) is obvious by (C_RefineL) and reflexivity of the compatibility relation. The case (2) is straightforward because T_i takes either of the form $z_1{}^F{:}\mathsf{int} \times \{z_2{}^F{:}\mathsf{int\ list} \mid e_0\}$ or $z_1{}^F{:}\mathsf{int} \times \{z_2{}^F{:}\tau\langle e'' \rangle \mid e_0\}$, and reflexivity of the compatibility relation and $\Sigma, \mathit{Trans}(F) \vdash \tau\langle e'' \rangle \parallel \mathsf{int\ list}$. □

5.3 Dynamic Correctness

Next, we show correctness of translation in the dynamic aspect: casts between refinement types with a translatable function F and the datatype generated from F succeed always. In particular, such casts convert "constructors" but not "structures". In this section, we assume that type definition environments include the datatype generated from a translatable function F.

Definition 9. *A constructor choice function δ is said to be trivial for τ when, if the type definition of τ takes the form $\mathsf{type}\,\tau_1\,\langle x{:}T\rangle = \overline{C_i \parallel D_i : T_i}^{\,i}$ and each D_i belongs to τ_2, then $\delta(\langle \tau_2\langle e_2\rangle \Leftarrow \tau_1\langle e_1\rangle\rangle^\ell\, C_i\langle e_3\rangle e_4) = D_i$ for any e_1, e_2, e_3, and e_4.*

We say that a constructor choice function is trivial when it is trivial for $\mathit{Trans}(F)$.

Lemma 58. *Let δ be a trivial choice function. Suppose that*

$$\mathit{Trans}(F) = \mathsf{type}\,\tau\,\langle y^F{:}T^F\rangle = D \parallel [\,] : \{z{:}\mathsf{unit} \mid {e_1}^F\} \mid \overline{D_i \parallel (::) : {z_1}^F{:}\mathsf{int} \times \{{z_2}^F{:}T_i \mid e_i\}}^{\,i}.$$

If $\varnothing \vdash \langle \mathsf{int\ list} \Leftarrow \tau\langle e\rangle\rangle^\ell\, v : \mathsf{int\ list}$ under δ, then $\langle \mathsf{int\ list} \Leftarrow \tau\langle e\rangle\rangle^\ell\, v \longrightarrow^* v'$ under δ for some v' which is obtained by replacing data constructor D and D_i of which v consists with $[\,]$ and $(::)$, respectively.

Proof. We proceed by structural induction on v. Since $\varnothing \vdash \langle \mathsf{int\ list} \Leftarrow \tau\langle e\rangle\rangle^\ell\, v : \mathsf{int\ list}$, we have $\varnothing \vdash \langle \mathsf{int\ list} \Leftarrow \tau\langle e\rangle\rangle^\ell : x{:}T_1' \to T_2'$ and $\varnothing \vdash v : T_1'$ and $T_2'\,\{v/x\} \equiv \mathsf{int\ list}$ for some x, T_1', and T_2' by Lemma 54. By Lemma 37, $T_2' = \mathsf{int\ list}$. By Lemmas 41 and 35, we have $\varnothing \vdash \tau\langle e\rangle$ and $\tau\langle e\rangle \equiv T_1'$. We perform case analysis on v by Lemmas 44 (4) and 43.

Case $v = D\langle e'\rangle v'$: Since δ is trivial, $\delta(\langle \mathsf{int\ list} \Leftarrow \tau\langle e\rangle\rangle^\ell\, D\langle e'\rangle v') = [\,]$. Thus, by (R_Datatype), (R_Forget) and (R_Base) with (E_Red),

$$\langle \mathsf{int\ list} \Leftarrow \tau\langle e\rangle\rangle^\ell\, D\langle e'\rangle v' \longrightarrow^* [\,].$$

Case $v = D_j\langle e'\rangle v'$: By Lemma 43, $\varnothing \vdash v' : {z_1}^F{:}\mathsf{int} \times \{{z_2}^F{:}T_i \mid e_i\}\,\{e'/y^F\}$. By Lemmas 44 (3) and 42, $v' = (v_1, v_2)$ for some v_1 and v_2 such that $\varnothing \vdash v_1 : \mathsf{int}$ and $\varnothing \vdash v_2 : \{{z_2}^F{:}T_i \mid e_i\}\,\{e'/y^F, v_1/{z_1}^F\}$. Note that e' is a closed term. Since δ is trivial, $\delta(\langle \mathsf{int\ list} \Leftarrow \tau\langle e\rangle\rangle^\ell\, D_j\langle e'\rangle v') = (::)$. Thus, by (R_Datatype), (R_Prod), (R_Base) and (R_Forget) with (E_Red),

$$\langle \mathsf{int\ list} \Leftarrow \tau\langle e\rangle\rangle^\ell\, D_j\langle e'\rangle v' \longrightarrow^* v_1 :: (\langle \mathsf{int\ list} \Leftarrow T_i\,\{e'/y^F, v_1/{z_1}^F\}\rangle^\ell\, v_2).$$

From *Trans*, there are two cases we have to consider. If $T_i = \mathsf{int\ list}$, then $\langle \mathsf{int\ list} \Leftarrow \tau\langle e\rangle\rangle^\ell\, D_j\langle e'\rangle v' \longrightarrow^* v_1 :: v_2$ by (R_DatatypeMono).. Otherwise, if $T_i = \tau\langle e''\rangle$ for some e'', then we finish by the IH, noting $\varnothing \vdash \langle \mathsf{int\ list} \Leftarrow T_i\,\{e'/y^F, v_1/{z_1}^F\}\rangle^\ell\, v_2 : \mathsf{int\ list}$, which follows from well-typedness of v_2, compatibility of $\mathsf{int\ list}$ and τ, (T_Cast), and (T_App). □

Definition 10 (Notation)**.** *Let σ be a (simultaneous) substitution. Then, we write $\sigma(e)$ to denote application of σ to e.*

Lemma 59. *Let F be a translatable function, v, v_1 and v_2 be closed values, σ be a simultaneous substitution including $\{F/f^F, v/y^F, v_1/{z_1}^F, v_2/{z_2}^F\}$, and e be a subterm of ${e_2}^F$. If $\sigma(e) \longrightarrow^* \mathsf{true}$, then there is a unique pair $(e_{\boldsymbol{opt}0}, e_0) \in \mathit{GenContracts}(e)$ such that*

- *$\sigma(e_0) \longrightarrow^* \mathsf{true}$ and*
- *for any e', $e_{\boldsymbol{opt}0} = \mathit{Some}\,e'$ implies $F\,\sigma(e')\,v_2 \longrightarrow^* \mathsf{true}$.*

Proof. By structural induction on e.

Case $e = \mathsf{true}$: Obvious since $\mathit{GenContracts}(\mathsf{true}) = \{(\mathit{None}, \mathsf{true})\}$.

Case $e = \mathsf{false}$: Contradictory; $\sigma(e) \longrightarrow^* \mathsf{false}$.

Case $e = \mathsf{if}\, f^F\, e'\, {z_2}^F\, \mathsf{then}\, e'_2\, \mathsf{else}\, e'_3$ where $\mathsf{FV}(e') \subseteq \{y^F, {z_1}^F\}$: By definition of *GenContracts*, we have

$$GenContracts(e) = \{(Some\, e', e'_2)\} \cup \{(e_{\mathbf{opt}_0}, \mathsf{if}\, f^F\, e'\, {z_2}^F\, \mathsf{then}\,\mathsf{false}\,\mathsf{else}\, e''_3) \mid (e_{\mathbf{opt}_0}, e''_3) \in GenContracts(e'_3)\}.$$

By case analysis on evaluation of $\sigma(f^F\, e'\, {z_2}^F) = F\,\sigma(e')\, v_2$. Note that the evaluation result is either true or false.

Case $F\,\sigma(e')\, v_2 \longrightarrow^* \mathsf{true}$: We have

$$\begin{array}{rcl}\sigma(\mathsf{if}\, f^F\, e'\, {z_2}^F\, \mathsf{then}\, e'_2\, \mathsf{else}\, e'_3) & \longrightarrow^* & \mathsf{if}\,\mathsf{true}\,\mathsf{then}\,\sigma(e'_2)\,\mathsf{else}\,\sigma(e'_3) \\ & \longrightarrow & \sigma(e'_2).\end{array}$$

Since $\sigma(e) \longrightarrow^* \mathsf{true}$, we find that $\sigma(e'_2) \longrightarrow^* \mathsf{true}$. Because

$$\begin{array}{rcl}\sigma(\mathsf{if}\, f^F\, e'\, {z_2}^F\, \mathsf{then}\,\mathsf{false}\,\mathsf{else}\, e''_3) & \longrightarrow^* & \mathsf{if}\,\mathsf{true}\,\mathsf{then}\,\mathsf{false}\,\mathsf{else}\,\sigma(e''_3) \\ & \longrightarrow & \mathsf{false},\end{array}$$

pair $(Some\, e', e'_2)$ is the unique one satisfying the property above.

Case $F\,\sigma(e')\, v_2 \longrightarrow^* \mathsf{false}$: We have

$$\begin{array}{rcl}\sigma(\mathsf{if}\, f^F\, e'\, {z_2}^F\, \mathsf{then}\, e'_2\, \mathsf{else}\, e'_3) & \longrightarrow^* & \mathsf{if}\,\mathsf{false}\,\mathsf{then}\,\sigma(e'_2)\,\mathsf{else}\,\sigma(e'_3) \\ & \longrightarrow & \sigma(e'_3).\end{array}$$

Since $\sigma(e) \longrightarrow^* \mathsf{true}$, we find that $\sigma(e'_3) \longrightarrow^* \mathsf{true}$. By the IH, there is a unique pair $(e_{\mathbf{opt}}, e''_3) \in GenContracts(e'_3)$ satisfying the above property. We have $\sigma(\mathsf{if}\, f^F\, e'\, {z_2}^F\, \mathsf{then}\,\mathsf{false}\,\mathsf{else}\, e''_3) \longrightarrow^* \mathsf{true}$. Since $F\,\sigma(e')\, v_2 \longrightarrow^* \mathsf{false}$, pair $(e_{\mathbf{opt}}, \mathsf{if}\, f^F\, e'\, {z_2}^F\, \mathsf{then}\,\mathsf{false}\,\mathsf{else}\, e''_3)$ is the unique one satisfying the property above.

Case $e = \mathsf{if}\, e'_1\, \mathsf{then}\, e'_2\, \mathsf{else}\, e'_3$ where $e'_1 \neq f^F\, e'\, {z_2}^F$ for any e' such that $\mathsf{FV}(e') \subseteq \{y^F, {z_1}^F\}$: By case analysis on evaluation of $\sigma(e'_1)$. Note that the evaluation result is either true or false.

Case $\sigma(e'_1) \longrightarrow^* \mathsf{true}$: Since $\sigma(\mathsf{if}\, e'_1\, \mathsf{then}\, e'_2\, \mathsf{else}\, e'_3) \longrightarrow^* \sigma(e'_2) \longrightarrow^* \mathsf{true}$, there a unique pair $(e_{\mathbf{opt}}, e''_2) \in GenContracts(e'_2)$ satisfying the above property, by the IH. Since $\sigma(\mathsf{if}\, e'_1\, \mathsf{then}\,\mathsf{false}\,\mathsf{else}\, e''_3) \longrightarrow^* \mathsf{false}$ for any e''_3, pair $(e_{\mathbf{opt}}, \mathsf{if}\, e'_1\, \mathsf{then}\, e''_2\, \mathsf{else}\,\mathsf{false})$ is the unique one satisfying the property above.

Case $\sigma(e'_1) \longrightarrow^* \mathsf{false}$: Since $\sigma(\mathsf{if}\, e'_1\, \mathsf{then}\, e'_2\, \mathsf{else}\, e'_3) \longrightarrow^* \sigma(e'_3) \longrightarrow^* \mathsf{true}$, there a unique pair $(e_{\mathbf{opt}}, e''_3) \in GenContracts(e'_3)$ satisfying the above property, by the IH. Since $\sigma(\mathsf{if}\, e'_1\, \mathsf{then}\, e''_2\, \mathsf{else}\,\mathsf{false}) \longrightarrow^* \mathsf{false}$ for any e''_2, pair $(e_{\mathbf{opt}}, \mathsf{if}\, e'_1\, \mathsf{then}\,\mathsf{false}\,\mathsf{else}\, e''_3)$ is the unique one satisfying the property above.

Case $e = \mathsf{match}\, e'_0\, \mathsf{with}\, \overline{C_i\, x_i \rightarrow e'_i}^{\,i \in \{1,\dots,n\}}$: Without loss of generality, we can suppose that each x_i is fresh for σ. Since $\sigma(e) \longrightarrow^* \mathsf{true}$, we find that $\sigma(e'_0) \longrightarrow^* C_j\langle e'\rangle v'$ for some $j \in \{1,\dots,n\}$, e' and v', and thus $\sigma(e'_j)\,\{v'/x_j\} \longrightarrow^* \mathsf{true}$. By the IH, there is a unique pair $(e_{\mathbf{opt}}, e''_j) \in GenContracts(e'_j)$ satisfying the above property. Since $\sigma(\mathsf{match}\, e'_0\, \mathsf{with}\, C_j\, x_j \rightarrow \mathsf{false} \mid \overline{C_i\, x_i \rightarrow e''_i}^{\,i \in \{1,\dots,n\}\setminus\{j\}}) \longrightarrow^* \mathsf{false}$, pair $(e_{\mathbf{opt}}, \mathsf{match}\, e'_0\, \mathsf{with}\, C_j\, x_j \rightarrow e''_j \mid \overline{C_i\, x_i \rightarrow \mathsf{false}}^{\,i \in \{1,\dots,n\}\setminus\{j\}})$ is the unique one satisfying the property above.

Case otherwise: Obvious because $Trans(e) = \{(None, e)\}$.

□

In what follows, we compute constructor choice functions to convert data structures. Before it, we show that extensions of constructor choice functions are conservative with respect to evaluation results.

Lemma 60. *Let δ' be an extension of constructor choice function δ. If $\delta \vdash e \longrightarrow^* v$, then $\delta' \vdash e \longrightarrow^* v$.*

Proof. From the two facts: (1) δ returns a constructor whenever taking cast applications in the evaluation $e \longrightarrow^* v$ and (2) δ' returns the same constructor as δ for cast applications contained by the domain of δ. □

Definition 11 (Notation)**.** *We write $\delta_1 \uplus \delta_2$ to denote the disjoint union of constructor choice functions δ_1 and δ_2.*

Theorem 2 (From Refinement Types to Datatypes)**.** *Suppose that*

$$\mathit{Trans}(F) = \mathsf{type}\,\tau\,\langle y^F{:}T^F\rangle = D \parallel [\,] : \{z{:}\mathsf{unit} \mid {e_1}^F\} \mid \overline{D_i \parallel (::) : {z_1}^F{:}\mathsf{int} \times \{{z_2}^F{:}T_i \mid e_i\}}^{\,i}.$$

Let δ be a trivial constructor choice function such that $\delta(\langle \tau\langle e'\rangle \Leftarrow \mathsf{int\ list}\rangle^\ell\, v')$ is undefined for any e' and sublist v' of v.

If $\varnothing \vdash \langle \tau\langle e\rangle \Leftarrow \{x{:}\mathsf{int\ list} \mid F\,e\,x\}\rangle^\ell\, v : \tau\langle e\rangle$ under δ, then there exists an extension δ' of δ such that $\langle \tau\langle e\rangle \Leftarrow \{l{:}\mathsf{int\ list} \mid F\,e\,l\}\rangle^\ell\, v \longrightarrow^* v'$ under δ' where v' is obtained by replacing some occurrences of data constructors $[\,]$ and $(::)$ of which v consists with D and one of $\overline{D_i}^{\,i}$, respectively.

Proof. By Lemma 54, we have $\varnothing \vdash \langle \tau\langle e\rangle \Leftarrow \{x{:}\mathsf{int\ list} \mid F\,e\,x\}\rangle^\ell : x_0{:}T_{01} \to T_{02}$ and $\varnothing \vdash v : T_{01}$ and $T_{02}\,\{v/x_0\} \equiv \tau\langle e\rangle$ for some x_0, T_{01} and T_{02}. By Lemmas 41 and 35 and (T_Conv), $\varnothing \vdash v : \{x{:}\mathsf{int\ list} \mid F\,e\,x\}$ and so $F\,e\,v \longrightarrow^* \mathsf{true}$ by Theorem 1 (noting that e is a closed term since since $\varnothing \vdash \tau\langle e\rangle$ by Lemma 46). Thus, $e \longrightarrow^* v'$ for some v'.

We proceed by case analysis on v by Lemmas 44 (4) and 43.

Case $v = [\,]$: Let $\delta' = \delta \uplus \{\langle \tau\langle e\rangle \Leftarrow \mathsf{int\ list}\rangle^\ell\,[\,] \mapsto D\}$. Then, by (R_Forget) and (R_Datatype) with (E_Red),

$$\delta' \vdash \langle \tau\langle e\rangle \Leftarrow \{x{:}\mathsf{int\ list} \mid F\,e\,x\}\rangle^\ell\,[\,] \longrightarrow^* D\langle e\rangle(\langle \{z{:}\mathsf{unit} \mid {e_1}^F\,\{e/x\}\} \Leftarrow \mathsf{unit}\rangle^\ell\,()).$$

Since $F\,v'\,v \longrightarrow^* \mathsf{true}$, we find that ${e_1}^F\,\{F/f^F, v'/y^F, v/x^F\} \longrightarrow^* \mathsf{true}$. Since F is translatable, we have $y{:}T \vdash {e_1}^F : \mathsf{Bool}$ and so ${e_1}^F\,\{F/f^F, v'/y^F, v/x^F\} = {e_1}^F\,\{v'/y^F\}$. Thus, ${e_1}^F\,\{v'/y^F\} \longrightarrow^* \mathsf{true}$. Since $e \Rrightarrow^* v'$ by Lemma 2, we have ${e_1}^F\,\{e/y^F\} \Rrightarrow^* {e_1}^F\,\{v'/y^F\}$ by Lemma 5 (2). By Lemma 30 (2), ${e_1}^F\,\{e/y^F\} \longrightarrow^* \mathsf{true}$. Since $\delta' \vdash {e_1}^F\,\{e/y^F\} \longrightarrow^* \mathsf{true}$ by Lemma 60, we have

$$\langle \tau\langle e\rangle \Leftarrow \{x{:}\mathsf{int\ list} \mid F\,e\,x\}\rangle^\ell\,[\,] \longrightarrow^* D\langle e\rangle().$$

by (R_PreCheck), (R_Base), (R_Check), and (R_OK) with (E_Red).

Case $v = (v_1 :: v_2)$: Since $F\,v'\,v \longrightarrow^* \mathsf{true}$, we find that

$${e_2}^F\,\{F/f^F, v'/y^F, v/x^F, v_1/{z_1}^F, v_2/{z_2}^F\} \longrightarrow^* \mathsf{true}.$$

Since F is translatable, $f^F{:}T^F \to \mathsf{int\ list} \to \mathsf{Bool}, y^F{:}T^F, {z_1}^F{:}\mathsf{int}, {z_2}^F{:}\mathsf{int\ list} \vdash {e_2}^F : \mathsf{Bool}$ and so

$${e_2}^F\,\{F/f^F, v'/y^F, v/x^F, v_1/{z_1}^F, v_2/{z_2}^F\} = {e_2}^F\,\{F/f^F, v'/y^F, v_1/{z_1}^F, v_2/{z_2}^F\}.$$

By Lemma 59, there is a unique pair $(e_{\mathbf{opt}_0}, e_0) \in \mathit{GenContracts}({e_2}^F)$ satisfying the property stated in Lemma 59. We perform case analysis on $e_{\mathbf{opt}_0}$.

Case $e_{\mathbf{opt}_0} = \mathit{Some}\,e'_0$: There exists some D_j such that

$$\mathit{CtrArgOf}(D_j) = {z_1}^F{:}\mathsf{int} \times T_j$$

where $T_j = \{{z_2}^F{:}\tau\langle e'_0\rangle \mid \mathsf{let}\,{z_2}^F = \langle \mathsf{int\ list} \Leftarrow \tau\langle e'_0\rangle\rangle^\ell\,{z_2}^F\,\mathsf{in}\,e_0\,\{F/f^F\}\}$. For any δ', if $\delta'(\langle \tau\langle e\rangle \Leftarrow \mathsf{int\ list}\rangle^\ell\,(v_1 :: v_2)) = D_j$, then by (R_Forget), (R_Datatype), (R_Prod), and (R_Base) with (E_Red),

$$\delta' \vdash \langle \tau\langle e\rangle \Leftarrow \{x{:}\mathsf{int\ list} \mid F\,e\,x\}\rangle^\ell\,(v_1 :: v_2) \longrightarrow^* D_j\langle e\rangle(v_1, \langle\!\langle T_j, \langle \tau\langle e'_0\rangle \Leftarrow \mathsf{int\ list}\rangle^\ell\,v_2\rangle\!\rangle^\ell\,\{e/y^F, v_1/{z_1}^F\}).$$

Let $e''_0 = e'_0\,\{e/y^F, v_1/{z_1}^F\}$. By Lemmas 56 and 33, we have $\varnothing \vdash e''_0 : T^F$ since $\varnothing \vdash v_1 : \mathsf{int}$ by Lemma 43, and $\varnothing \vdash e : T^F$ from inversion of $\varnothing \vdash \tau\langle e\rangle$. Thus, $x{:}\mathsf{int\ list} \vdash F\,e''_0\,x : \mathsf{Bool}$ by Lemma 32, (T_Var) and (T_App), and so $\varnothing \vdash \{x{:}\mathsf{int\ list} \mid F\,e''_0\,x\}$ by (WT_Refine).

Since $e \longrightarrow^* v'$, we have $F\,e''_0\,v_2 \Rrightarrow^* F\,e'_0\,\{v'/y^F, v_1/{z_1}^F\}\,v_2$ by Lemmas 2 and 5 (2). Since $F\,e'_0\,\{v'/y^F, v_1/{z_1}^F\}\,v_2 \longrightarrow^* \mathsf{true}$ by Lemma 59, we have $F\,e''_0\,v_2 \longrightarrow^* \mathsf{true}$ by Lemma 30 (2). Thus, by (T_Exact), $\varnothing \vdash v_2 : \{x{:}\mathsf{int\ list} \mid F\,e''_0\,x\}$ since $\varnothing \vdash v_2 : \mathsf{int\ list}$ by Lemma 43. Since $\tau\langle e''_0\rangle \parallel$

$\{x{:}\mathsf{int\ list} \mid F\, e_0''\, x\}$ by (C_Datatype) and (C_RefineL) (noting the compatibility relation is a equivalence one), and $\varnothing \vdash \tau\langle e_0''\rangle$ by (WT_Datatype), we have

$$\varnothing \vdash \langle \tau\langle e_0''\rangle \Leftarrow \{x{:}\mathsf{int\ list} \mid F\, e_0''\, x\}\rangle^{\ell}\, v_2 : \tau\langle e_0''\rangle$$

by (T_Cast) and (T_App). By the IH, there exist some δ'' and v_2' such that

$$\delta'' \vdash \langle \tau\langle e_0''\rangle \Leftarrow \{x{:}\mathsf{int\ list} \mid F\, e_0''\, x\}\rangle^{\ell}\, v_2 \longrightarrow^{*} v_2'$$

and δ'' is an extension of δ, and v_2' is obtained by replacing data constructor $[]$ and $(::)$ of which v_2 consists with D and one of $\overline{D_i}^{\,i}$, respectively. Let $\delta''' = \{\langle \tau\langle e\rangle \Leftarrow \mathsf{int\ list}\rangle^{\ell}\, (v_1 :: v_2) \mapsto D_j\} \uplus \delta''$. Then,

$$\delta''' \vdash \langle \tau\langle e\rangle \Leftarrow \{x{:}\mathsf{int\ list} \mid F\, e\, x\}\rangle^{\ell}\, (v_1 :: v_2) \longrightarrow^{*} D_j\langle e\rangle(v_1, \langle\!\langle T_j\, \{e/y^F, v_1/{z_1}^F\}, v_2'\rangle\!\rangle^{\ell}).$$

Since $\varnothing \vdash v_2' : \tau\langle e_0''\rangle$ by Theorem 1, we have $\varnothing \vdash \langle \mathsf{int\ list} \Leftarrow \tau\langle e_0''\rangle\rangle^{\ell}\, v_2' : \mathsf{int\ list}$ by (T_Cast) and (T_App). By Lemma 58, we have $\langle \mathsf{int\ list} \Leftarrow \tau\langle e_0''\rangle\rangle^{\ell}\, v_2' \longrightarrow^{*} v_2$ since δ is trivial. Since $e_0\, \{F/f^F, v'/y^F, v_1/{z_1}^F, v_2/{z_2}^F\}$ true by Lemma 59, we have $e_0\, \{F/f^F, e/y^F, v_1/{z_1}^F, v_2/{z_2}^F\} \longrightarrow^{*} \mathsf{true}$ by Lemmas 2, 5 (2) and 30 (2). Thus,

$$(\mathsf{let}\, {z_2}^F = \langle \mathsf{int\ list} \Leftarrow \tau\langle e_0'\rangle\rangle^{\ell}\, {z_2}^F\, \mathsf{in}\, e_0\, \{F/f^F\})\, \{e/y^F, v_1/{z_1}^F, v_2'/{z_2}^F\} \longrightarrow^{*} \mathsf{true}.$$

Therefore, by (R_Check) and (R_OK) with (E_Red) and Lemma 60,

$$\delta''' \vdash \langle \tau\langle e\rangle \Leftarrow \{x{:}\mathsf{int\ list} \mid F\, e\, x\}\rangle^{\ell}\, (v_1 :: v_2) \longrightarrow^{*} D_j\langle e\rangle(v_1, v_2').$$

Case $e_{\mathbf{opt}_0} = \mathit{None}$: There exists some D_j such that

$$\mathit{CtrArgOf}(D_j) = {z_1}^F{:}\mathsf{int} \times T_j$$

where $T_j = \{{z_2}^F{:}\mathsf{int\ list} \mid e_0\, \{F/f^F\}\}$. Let $\delta' = \delta \uplus \{\langle \tau\langle e\rangle \Leftarrow \mathsf{int\ list}\rangle^{\ell}\, (v_1 :: v_2) \mapsto D_j\}$. By (R_Forget), (R_Datatype), (R_Prod), (R_Base) with (E_Red),

$$\delta' \vdash \langle \tau\langle e\rangle \Leftarrow \{x{:}\mathsf{int\ list} \mid F\, e\, x\}\rangle^{\ell}\, (v_1 :: v_2) \longrightarrow^{*} D_j\langle e\rangle(v_1, \langle\!\langle T_j, \langle \mathsf{int\ list} \Leftarrow \mathsf{int\ list}\rangle^{\ell}\, v_2\rangle\!\rangle^{\ell}\, \{e/y^F, v_1/{z_1}^F\}).$$

Since $\langle \mathsf{int\ list} \Leftarrow \mathsf{int\ list}\rangle^{\ell}\, v_2 \longrightarrow^{*} v_2$ by (R_DatatypeMono), we have

$$\delta' \vdash \langle \tau\langle e\rangle \Leftarrow \{x{:}\mathsf{int\ list} \mid F\, e\, x\}\rangle^{\ell}\, (v_1 :: v_2) \longrightarrow^{*} D_j\langle e\rangle(v_1, \langle T_j, e_0\, \{F/f^F, v_2/{z_2}^F\}, v_2\rangle^{\ell}\, \{e/y^F, v_1/{z_1}^F\})$$

by (E_Red)/(R_Check). Since $e_0\, \{F/f^F, v'/y^F, v_1/{z_1}^F, v_2/{z_2}^F\} \longrightarrow^{*} \mathsf{true}$ by Lemma 59, we have $e_0\, \{F/f^F, e/y^F, v_1/{z_1}^F, v_2/{z_2}^F\} \longrightarrow^{*} \mathsf{true}$ by Lemmas 2, 5 (2) and 30 (2). Thus, by (E_Red)/(R_OK) and Lemma 60,,

$$\delta' \vdash \langle \tau\langle e\rangle \Leftarrow \{x{:}\mathsf{int\ list} \mid F\, e\, x\}\rangle^{\ell}\, (v_1 :: v_2) \longrightarrow^{*} D_j\langle e\rangle(v_1, v_2).$$

□

Lemma 61. *Let F be a translatable function, e be a subterm of ${e_2}^F$, and σ be a simultaneous substitution including $\{F/f^F, e'/y^F, v_1/{z_1}^F, v_2/{z_2}^F\}$. If $(e_{\mathbf{opt}_0}, e_0) \in \mathit{GenContracts}(e)$ and $\sigma(e_0) \longrightarrow^{*} \mathsf{true}$ and $e_{\mathbf{opt}_0} = \mathit{Some}\, e''$ implies $F\, \sigma(e'')\, v_2 \longrightarrow^{*} \mathsf{true}$ for any e'', then $\sigma(e) \longrightarrow^{*} \mathsf{true}$.*

Proof. By structural induction on e.

Case $e = \mathsf{true}$: Obvious.

Case false: Contradictory; $\mathit{GenContracts}(\mathsf{false}) = \varnothing$.

Case $e = \mathsf{if}\, f^F\, e''\, {z_2}^F\, \mathsf{then}\, e_2'\, \mathsf{else}\, e_3'$ where $\mathsf{FV}(e'') \subseteq \{y^F, {z_1}^F\}$: There are two cases which we have to consider by case analysis on e_0.

Case $e_0 = e_2'$: Since $e_{\mathbf{opt}_0} = \mathit{Some}\, e''$, we have $F\, \sigma(e'')\, v_2 \longrightarrow^{*} \mathsf{true}$. Thus, $\sigma(\mathsf{if}\, f^F\, e''\, {z_2}^F\, \mathsf{then}\, e_0\, \mathsf{else}\, e_3') \longrightarrow^{*} \sigma(e_0) \longrightarrow^{*} \mathsf{true}$.

Case $e_0 = \mathsf{if}\, f^F\, e''\, {z_2}^F\, \mathsf{then\,false\,else}\, e''_3$ where $(e_{\mathbf{opt}_0}, e''_3) \in \mathit{GenContracts}(e'_3)$: Since $\sigma(e_0) \longrightarrow^* \mathsf{true}$, we find that $F\,\sigma(e'')\,v_2 \longrightarrow^* \mathsf{false}$ and $\sigma(e''_3) \longrightarrow^* \mathsf{true}$. Since $(e_{\mathbf{opt}_0}, e''_3) \in \mathit{GenContracts}(e'_3)$, we have $\sigma(e'_3) \longrightarrow^* \mathsf{true}$ by the IH. Thus, $\sigma(\mathsf{if}\, f^F\, e''\, {z_2}^F\, \mathsf{then}\, e'_2\, \mathsf{else}\, e'_3) \longrightarrow^* \mathsf{true}$.

Case $e = \mathsf{if}\, e'_1\, \mathsf{then}\, e'_2\, \mathsf{else}\, e'_3$ where $e'_1 \neq f^F\, e''\, {z_2}^F$ for any e'' such that $\mathsf{FV}(e'') \subseteq \{y^F, {z_1}^F\}$: There are two cases which we have to consider by case analysis on e_0.

Case $e_0 = \mathsf{if}\, e'_1\, \mathsf{then}\, e''_2\, \mathsf{else\,false}$ where $(e_{\mathbf{opt}_0}, e''_2) \in \mathit{GenContracts}(e'_2)$: Since $\sigma(e_0) \longrightarrow^* \mathsf{true}$, we find that $\sigma(e'_1) \longrightarrow^* \mathsf{true}$ and $\sigma(e''_2) \longrightarrow^* \mathsf{true}$. Since $(e_{\mathbf{opt}_0}, e''_2) \in \mathit{GenContracts}(e'_2)$, we have $\sigma(e'_2) \longrightarrow^* \mathsf{true}$ by the IH. Thus, $\sigma(\mathsf{if}\, e'_1\, \mathsf{then}\, e'_2\, \mathsf{else}\, e'_3) \longrightarrow^* \mathsf{true}$.

Case $e_0 = \mathsf{if}\, e'_1\, \mathsf{then\,false\,else}\, e''_3$ where $(e_{\mathbf{opt}_0}, e''_3) \in \mathit{GenContracts}(e'_3)$: Since $\sigma(e_0) \longrightarrow^* \mathsf{true}$, we find that $\sigma(e'_1) \longrightarrow^* \mathsf{false}$ and $\sigma(e''_3) \longrightarrow^* \mathsf{true}$. Since $(e_{\mathbf{opt}_0}, e''_3) \in \mathit{GenContracts}(e'_3)$, we have $\sigma(e'_3) \longrightarrow^* \mathsf{true}$ by the IH. Thus, $\sigma(\mathsf{if}\, e'_1\, \mathsf{then}\, e'_2\, \mathsf{else}\, e'_3) \longrightarrow^* \mathsf{true}$.

Case $e = \mathsf{match}\, e'_0\, \mathsf{with}\, \overline{C_i\, x_i \to e'_i}^{i \in \{1,\dots,n\}}$: For some j, we have $e_0 = \mathsf{match}\, e'_0\, \mathsf{with}\, C_j\, x_j \to e''_j \mid \overline{C_i\, x_i \to \mathsf{false}}^{i \in \{1,\dots,n\} \setminus \{j\}}$ where $(e_{\mathbf{opt}_0}, e''_j) \in \mathit{GenContracts}(e'_j)$. Since $\sigma(e_0) \longrightarrow^* \mathsf{true}$, we have $\sigma(e'_0) \longrightarrow^* C_j\langle e''\rangle v'$ and $\sigma(e''_j)\,\{v'/x_j\} \longrightarrow^* \mathsf{true}$ for some e'' and v'. By the IH, $\sigma(e'_j)\,\{v'/x_j\} \longrightarrow^* \mathsf{true}$. Thus, $\sigma(\mathsf{match}\, e'_0\, \mathsf{with}\, \overline{C_i\, x_i \to e'_i}^{i \in \{1,\dots,n\}}) \longrightarrow^* \mathsf{true}$.

Case otherwise: Obvious since $\mathit{GenContracts}(e) = \{(\mathit{None}, e)\}$. $\square$

Definition 12 (Termination)**.** *A closed term e terminates at a value, written as $e\downarrow$, if $e \longrightarrow^* v$ for some v. We say that argument terms to datatype τ in v terminate at values, written $v\downarrow_\tau$ when, for any E, $C \in \mathit{CtrsOf}(\tau)$, e_1 and v_2, if $v = E[C\langle e_1\rangle v_2]$, then $e_1\downarrow$.*

Lemma 62. *Let F be a translatable function and δ be a trivial constructor choice function. If $v\downarrow_\tau$ and $\varnothing \vdash \langle \mathsf{int\ list} \Leftarrow \tau\langle e\rangle\rangle^\ell\, v : \mathsf{int\ list}$, then $F\,e\,(\langle \mathsf{int\ list} \Leftarrow \tau\langle e\rangle\rangle^\ell\, v) \longrightarrow^* \mathsf{true}$.*

Proof. By structural induction on v. Suppose that

$$\mathit{Trans}(F) = \mathsf{type}\,\tau\,\langle y^F{:}T^F\rangle = D \parallel [\,] : \{z{:}\mathsf{unit} \mid {e_1}^F\} \mid \overline{D_i \parallel (::) : {z_1}^F{:}\mathsf{int} \times \{{z_2}^F{:}T_i \mid e_i\}}^{\,i}.$$

By Lemmas 54 and 41 and (T_Conv), we have $\varnothing \vdash v : \tau\langle e\rangle$. By Lemmas 44 (4) and 43, there are two cases which we have to consider by case analysis on v.

Case $v = D\langle e'\rangle v'$: Since $v\downarrow_\tau$, $e' \longrightarrow^* v''$ for some v''. By Lemmas 43 and 37, we have $\varnothing \vdash v' : \{z{:}\mathsf{unit} \mid {e_1}^F\,\{e'/y^F\}\}$ and $e' \equiv e$. By Theorem 1, we find that ${e_1}^F\,\{e'/y^F, v'/z\} = {e_1}^F\,\{e'/y^F\} \longrightarrow^* \mathsf{true}$. Since $\langle \mathsf{int\ list} \Leftarrow \tau\langle e\rangle\rangle^\ell\, D\langle e'\rangle v' \longrightarrow^* [\,]$ by Lemma 58, we have

$$\begin{array}{rcll} F\,e\,(\langle \mathsf{int\ list} \Leftarrow \tau\langle e\rangle\rangle^\ell\, v) & \equiv & F\,e'\,(\langle \mathsf{int\ list} \Leftarrow \tau\langle e\rangle\rangle^\ell\, v) & \text{(by Lemmas 1 and 5 (3))} \\ & \longrightarrow^* & F\,v''\,(\langle \mathsf{int\ list} \Leftarrow \tau\langle e\rangle\rangle^\ell\, v) & \\ & \longrightarrow^* & F\,v''\,[\,] & \\ & \longrightarrow^* & {e_1}^F\,\{v''/y^F\} & \\ & \equiv & {e_1}^F\,\{e'/y^F\}. & \text{(by Lemmas 2, 5 (3) and 1)} \end{array}$$

Thus, by Lemma 31 (2),

$$F\,e\,(\langle \mathsf{int\ list} \Leftarrow \tau\langle e\rangle\rangle^\ell\, v) \longrightarrow^* \mathsf{true}.$$

Case $v = D_j\langle e'\rangle v'$: By definition of *Trans*, there is a unique pair $(e_{\mathbf{opt}_0}, e_0) \in \mathit{GenContracts}({e_2}^F)$ such that $\mathit{CtrArgOf}(D_j)$ is constructed from the pair. By case analysis on $e_{\mathbf{opt}_0}$.

Case $e_{\mathbf{opt}_0} = \mathit{Some}\, e'_0$: We have

$$\mathit{CtrArgOf}(D_j) = {z_1}^F{:}\mathsf{int} \times \{{z_2}^F{:}\tau\langle e'_0\rangle \mid \mathsf{let}\, {z_2}^F = \langle \mathsf{int\ list} \Leftarrow \tau\langle e'_0\rangle\rangle^\ell\, {z_2}^F\, \mathsf{in}\, e_0\,\{F/f^F\}\}.$$

By Lemmas 43, 44 (3), 42 and 37, we have $v' = (v_1, v_2)$ and $\varnothing \vdash v_1 : \mathsf{int}$ and $\varnothing \vdash v_2 : \{{z_2}^F{:}\tau\langle e'_0\rangle \mid \mathsf{let}\, {z_2}^F = \langle \mathsf{int\ list} \Leftarrow \tau\langle e'_0\rangle\rangle^\ell\, {z_2}^F\, \mathsf{in}\, e_0\,\{F/f^F\}\}\,\{e'/y^F, v_1/{z_1}^F\}$ and $e \equiv e'$ for some v_1 and v_2. By Lemma 46, we have $\varnothing \vdash e : T^F$. Since $y^F{:}T^F, {z_1}^F{:}\mathsf{int} \vdash e'_0 : T^F$ by Lemma 56, we have $\varnothing \vdash e'_0\,\{e/y^F, v_1/{z_1}^F\} : T^F$. Since $\varnothing \vdash \tau\langle e'_0\rangle\,\{e/y^F, v_1/{z_1}^F\}$ by Lemmas 57 and 33 (2) and (T_Forget), we have

$$\varnothing \vdash v_2 : \tau\langle e'_0\rangle\,\{e/y^F, v_1/{z_1}^F\}$$

by Lemma 5 (3), (T_Forget), and (T_Conv). Thus, we have $\varnothing \vdash \langle \mathsf{int\ list} \Leftarrow \tau\langle e'_0\,\{e/y^F, v_1/{z_1}^F\}\rangle\rangle^\ell\, v_2 : \mathsf{int\ list}$ by (T_Forget), (T_Cast) and (T_App). By Lemma 58, there exists some v'_2 such that

$$\langle \mathsf{int\ list} \Leftarrow \tau\langle e'_0\,\{e/y^F, v_1/{z_1}^F\}\rangle\rangle^\ell\, v_2 \longrightarrow^* v'_2.$$

By the IH, we have

$$F\, e'_0\,\{e/y^F, v_1/{z_1}^F\}\,(\langle \mathsf{int\ list} \Leftarrow \tau\langle e'_0\,\{e/y^F, v_1/{z_1}^F\}\rangle\rangle^\ell\, v_2) \longrightarrow^* \mathsf{true}.$$

Thus, there exists some v'_0 such that $e'_0\,\{e/y^F, v_1/{z_1}^F\} \longrightarrow^* v'_0$ and $F\, v'_0\, v'_2 \longrightarrow^* \mathsf{true}$. Since $F\, e'_0\,\{e/y^F, v_1/{z_1}^F\}\, v'_2 \Rrightarrow^* F\, v'_0\, v'_2$ by Lemmas 2 and 5 (2), we have

$$F\, e'_0\,\{e/y^F, v_1/{z_1}^F\}\, v'_2 \longrightarrow^* \mathsf{true}$$

by Lemma 30 (2). By applying Lemma 51 to v_2, we have $e_0\,\{F/f^F, e'/y^F, v_1/{z_1}^F, v'_2/{z_2}^F\} \longrightarrow^* \mathsf{true}$. Thus, by Lemmas 5 (3) and 31, we have

$$e_0\,\{F/f^F, e/y^F, v_1/{z_1}^F, v'_2/{z_2}^F\} \longrightarrow^* \mathsf{true}.$$

By Lemma 61,

$${e_2}^F\,\{F/f^F, e/y^F, v_1/{z_1}^F, v'_2/{z_2}^F\} \longrightarrow^* \mathsf{true}.$$

Since $e' \longrightarrow^* v''$ for some v'' from $v\downarrow_\tau$, we have $v'' \equiv e$. By Lemmas 5 (3) and 31,

$${e_2}^F\,\{F/f^F, v''/y^F, v_1/{z_1}^F, v'_2/{z_2}^F\} \longrightarrow^* \mathsf{true}.$$

Thus,

$$\begin{array}{ll} & F\, e'\,(\langle \mathsf{int\ list} \Leftarrow \tau\langle e\rangle\rangle^\ell\, D_j\langle e'\rangle v') \\ \longrightarrow^* & F\, v''\,(\langle \mathsf{int\ list} \Leftarrow \tau\langle e\rangle\rangle^\ell\, D_j\langle e'\rangle v') \\ \longrightarrow^* & F\, v''\,(v_1 :: (\langle \mathsf{int\ list} \Leftarrow \tau\langle e'_0\,\{e/y^F, v_1/{z_1}^F\}\rangle\rangle^\ell\, v_2)) \\ \longrightarrow^* & F\, v''\,(v_1 :: v'_2) \\ \longrightarrow^* & {e_2}^F\,\{F/f^F, v''/y^F, v_1/{z_1}^F, v'_2/{z_2}^F\} \\ \longrightarrow^* & \mathsf{true}. \end{array}$$

Case $e_{\mathbf{opt}_0} = \mathit{None}$: We have $\mathit{CtrArgOf}(D_j) = {z_1}^F{:}\mathsf{int} \times \{{z_2}^F{:}\mathsf{int\ list} \mid e_0\,\{F/f^F\}\}$. By Lemmas 43, 44 (3), 42 and 37, we have $\varnothing \vdash e' : T^F$ and $v' = (v_1, v_2)$ and $\varnothing \vdash v_1 : \mathsf{int}$ and $\varnothing \vdash v_2 : \{{z_2}^F{:}\mathsf{int\ list} \mid e_0\,\{F/f^F\}\}\,\{e'/y^F, v_1/{z_1}^F\}$ for some v_1 and v_2. By Lemma 51, $e_0\,\{F/f^F, e'/y^F, v_1/{z_1}^F, v_2/{z_2}^F\} \longrightarrow^* \mathsf{true}$. By Lemma 61, we have ${e_2}^F\,\{F/f^F, e'/y^F, v_1/{z_1}^F, v_2/{z_2}^F\} \longrightarrow^* \mathsf{true}$. Since $e' \longrightarrow^* v''$ for some v'' from $v\downarrow_\tau$, we have ${e_2}^F\,\{F/f^F, v''/y^F, v_1/{z_1}^F, v_2/{z_2}^F\} \longrightarrow^* \mathsf{true}$ by Lemmas 2, 5 (2) and 30 (1). Thus,

$$F\, e'\,(\langle \mathsf{int\ list} \Leftarrow \tau\langle e\rangle\rangle^\ell\, D_j\langle e'\rangle v') \longrightarrow^* F\, v''\,(v_1 :: v_2) \longrightarrow^* \mathsf{true}.$$

$\square$

Theorem 3 (From Datatypes to Refinement Types)**.** *Suppose that*

$$\mathit{Trans}(F) = \mathsf{type}\,\tau\,\langle y^F{:}T^F\rangle = D \parallel [\,] : \{z{:}\mathsf{unit} \mid {e_1}^F\} \mid \overline{D_i \parallel (::) : {z_1}^F{:}\mathsf{int} \times \{{z_2}^F{:}T_i \mid e_i\}}^{\,i}.$$

Let δ be a trivial constructor choice function.

If $v\downarrow_\tau$ and $\varnothing \vdash v : \tau\langle e\rangle$, then $\langle \{x{:}\mathsf{int\ list} \mid F\, e\, x\} \Leftarrow \tau\langle e\rangle\rangle^\ell\, v \longrightarrow^* v'$ for some v' obtained by replacing data constructor D and D_i in v with $[\,]$ and $(::)$, respectively.

Proof. Since $\varnothing \vdash \tau\langle e\rangle$ Lemma 46 and $\mathsf{int\ list} \parallel \tau\langle e\rangle$, we have $\varnothing \vdash \langle \mathsf{int\ list} \Leftarrow \tau\langle e\rangle\rangle^\ell\, v : \mathsf{int\ list}$ by (T_Cast) and (T_App). By Lemma 58, $\langle \mathsf{int\ list} \Leftarrow \tau\langle e\rangle\rangle^\ell\, v \longrightarrow^* v'$ for some v' which satisfies the property in the statement above. By Lemma 62, we have $F\, e\,(\langle \mathsf{int\ list} \Leftarrow \tau\langle e\rangle\rangle^\ell\, v) \longrightarrow^* \mathsf{true}$. Thus, letting v'' be a value such that $e \longrightarrow^* v''$, we find that $F\, v''\, v' \longrightarrow^* \mathsf{true}$. By Lemmas 2, 5 (2) and 30 (2), $F\, e\, v' \longrightarrow^* \mathsf{true}$. Thus, by (R_PreCheck) and (R_OK) with (E_Red),

$$\langle \{x{:}\mathsf{int\ list} \mid F\, e\, x\} \Leftarrow \tau\langle e\rangle\rangle^\ell\, v \longrightarrow^* v'.$$

$\square$