A Appendix

A.1 Proofs

Lemma 1. *If* $\mathit{mbody}(\mathtt{m}, \mathrm{I}_d, \mathrm{I}_s) = (\mathrm{J}, \overline{\mathrm{I}_x}\ \overline{\mathtt{x}}, \mathrm{I}_e\ \mathtt{e}_0)$*, then* $\overline{\mathtt{x}} : \overline{\mathrm{I}_x}, \mathit{this} : \mathrm{J} \vdash \mathtt{e}_0 : \mathrm{I}_0$ *for some* $\mathrm{I}_0 <: \mathrm{I}_e$*.*

Proof. By the definition of `mbody`, the target method $\mathtt{m}$ is found in J. By the method typing rule (T-METHOD), there exists some $\mathrm{I}_0 <: \mathrm{I}_e$ such that $\overline{\mathtt{x}} : \overline{\mathrm{I}_x}, \mathtt{this} : \mathrm{J} \vdash \mathtt{e}_0 : \mathrm{I}_0$. □

Lemma 2 (Weakening). *If* $\Gamma \vdash \mathtt{e} : \mathrm{I}$*, then* $\Gamma, \mathtt{x} : \mathrm{J} \vdash \mathtt{e} : \mathrm{I}$*.*

Proof. Straightforward induction. □

Lemma 3 (Method Type Preservation). *If* $\mathit{mbody}(\mathtt{m}, \mathrm{J}, \mathrm{J}) = (\mathsf{K}, \overline{\mathrm{I}_x\ \_}, \mathrm{I}_e\ \_)$*, then for any* $\mathrm{I} <: \mathrm{J}$*,* $\mathit{mbody}(\mathtt{m}, \mathrm{I}, \mathrm{J}) = (\mathsf{K}', \overline{\mathrm{I}_x\ \_}, \mathrm{I}_e\ \_)$*.*

Proof. Since $\mathtt{mbody}(\mathtt{m}, \mathrm{J}, \mathrm{J})$ is defined, by (T-INTF) we derive that $\mathtt{mbody}(\mathtt{m}, \mathrm{I}, \mathrm{J})$ is also defined. Suppose that

$$\mathtt{findOrigin}(\mathtt{m}, \mathrm{J}, \mathrm{J}) = \{\mathrm{I}_0\}$$

$$\mathtt{findOverride}(\mathtt{m}, \mathrm{J}, \mathrm{I}_0) = \{\mathsf{K}\}$$

$$\mathtt{findOrigin}(\mathtt{m}, \mathrm{I}, \mathrm{J}) = \{\mathrm{I}_0'\}$$

$$\mathtt{findOverride}(\mathtt{m}, \mathrm{I}, \mathrm{I}_0') = \{\mathsf{K}'\}$$

Below we use $\mathrm{I}[\mathtt{m} \uparrow \mathrm{J}]$ to denote the type of method $\mathtt{m}$ defined in I that overrides J. We have to prove that $\mathsf{K}'[\mathtt{m} \uparrow \mathrm{I}_0'] = \mathsf{K}[\mathtt{m} \uparrow \mathrm{I}_0]$. Two facts:

– A. By (T-INTF), `canOverride` ensures that an override between any two original methods preserves the method type. Formally,

$$\mathrm{I}_1 <: \mathrm{I}_2 \quad \Rightarrow \quad \mathrm{I}_1[\mathtt{m} \uparrow \mathrm{I}_1] = \mathrm{I}_2[\mathtt{m} \uparrow \mathrm{I}_2]$$

– B. By (T-METHOD) and (T-ABSMETHOD), any partial override also preserves method type. Formally,

$$\mathrm{I}_1 <: \mathrm{I}_2 \quad \Rightarrow \quad \mathrm{I}_1[\mathtt{m} \uparrow \mathrm{I}_2] = \mathrm{I}_2[\mathtt{m} \uparrow \mathrm{I}_2]$$

By definition of `findOverride`, $\mathsf{K} <: \mathrm{I}_0, \mathsf{K}' <: \mathrm{I}_0'$. By Fact B,

$$\mathsf{K}[\mathtt{m} \uparrow \mathrm{I}_0] = \mathrm{I}_0[\mathtt{m} \uparrow \mathrm{I}_0] \quad \mathsf{K}'[\mathtt{m} \uparrow \mathrm{I}_0'] = \mathrm{I}_0'[\mathtt{m} \uparrow \mathrm{I}_0']$$

Hence it suffices to prove that $\mathrm{I}_0'[\mathtt{m} \uparrow \mathrm{I}_0'] = \mathrm{I}_0[\mathtt{m} \uparrow \mathrm{I}_0]$. Actually when calculating $\mathtt{findOrigin}(\mathtt{m}, \mathrm{J}, \mathrm{J})$, by the definition of `findOrigin` we know that $\mathrm{I}_0 <: \mathrm{J}$ and $\mathrm{I}_0[\mathtt{m}\ \mathtt{override}\ \mathrm{I}_0]$ is defined. So when calculating $\mathtt{findOrigin}(\mathtt{m}, \mathrm{I}, \mathrm{J})$ with $\mathrm{I} <: \mathrm{J}$, I_0 should also appear in the set before pruned, since the conditions are again satisfied. But after pruning, only I_0' is obtained, by definition of `prune` it implies $\mathrm{I}_0' <: \mathrm{I}_0$. By Fact A, the proof is done.

□

Lemma 4 (Term Substitution Preserves Typing). *If* $\Gamma, \overline{x} : \overline{I_x} \vdash e : I$*, and* $\Gamma \vdash \overline{y} : \overline{I_x}$*, then* $\Gamma \vdash [\overline{y}/\overline{x}]e : I$.

Proof. We prove by induction. The expression e has the following cases:

Case Var. Let $e = x$. If $x \notin \overline{x}$, then the substitution does not change anything. Otherwise, since $\overline{y}$ have the same types as $\overline{x}$, it immediately finishes the case.

Case Invk. Let $e = e_0.m(\overline{e})$. By (T-INVK) we can suppose that

$$\Gamma, \overline{x} : \overline{I_x} \vdash e_0 : I_0 \quad \texttt{mbody}(m, I_0, I_0) = (\_, \overline{J\ \_}, I\ \_)$$

$$\Gamma, \overline{x} : \overline{I_x} \vdash \overline{e} : \overline{I_e} \quad \overline{I_e} <: \overline{J} \quad \Gamma, \overline{x} : \overline{I_x} \vdash e : I$$

By induction hypothesis,

$$\Gamma \vdash [\overline{y}/\overline{x}]e_0 : I_0 \quad \Gamma \vdash [\overline{y}/\overline{x}]\overline{e} : \overline{I_e}$$

Again by (T-INVK), $\Gamma \vdash [\overline{y}/\overline{x}]e : I$.

Case New. Straightforward.

Case Anno. Straightforward by induction hypothesis and (T-ANNO). □

Proof for Theorem 1

Proof.

Case S-Invk. Let

$$e = ((J)\texttt{new}\ I()).m(\overline{v}) \quad \Gamma \vdash e : I_e$$

$$e' = (I_{e_0})[\overline{(I_x)v}/\overline{x}, (I_0)\texttt{new}\ I()/\texttt{this}]e_0$$

$$\texttt{mbody}(m, I, J) = (I_0, \overline{I_x\ x}, I_{e_0}\ e_0)$$

Since $\texttt{mbody}(m, I, J)$ is defined, the definition of $\texttt{mbody}$ ensures that $I <: J$. And since e is well-typed, by (T-INVK),

$$\Gamma \vdash \overline{v} : \overline{I_v} \quad \overline{I_v} <: \overline{I_x}$$

By the rules (T-ANNO) and (T-NEW),

$$\Gamma \vdash \overline{(I_x)v} : \overline{I_x} \quad \Gamma \vdash (I_0)\texttt{new}\ I() : I_0$$

On the other hand, by Lemma **??**,

$$\overline{x} : \overline{I_x}, \texttt{this} : I_0 \vdash e_0 : I'_{e_0} \quad I'_{e_0} <: I_{e_0}$$

By Lemma **??**,

$$\Gamma, \overline{x} : \overline{I_x}, \texttt{this} : I_0 \vdash e_0 : I'_{e_0}$$

Hence by Lemma **??**, the substitution preserves typing, thus

$$\Gamma \vdash [\overline{(I_x)v}/\overline{x}, (I_0)\texttt{new}\ I()/\texttt{this}]e_0 : I'_{e_0}$$

Since $I'_{e_0} <: I_{e_0}$, the conditions of (T-ANNO) are satisfied, hence $\Gamma \vdash e' : I_{e_0}$. Now we only need to prove that $I_{e_0} = I_e$. Since I_{e_0} is from $\texttt{mbody}(m, I, J)$, whereas I_e is from $\texttt{mbody}(m, J, J)$, by the rule (T-INVK) on e. Since $I <: J$, by Lemma **??**, $I_{e_0} = I_e$.

Case C-Receiver. Straightforward induction.

Case C-Args. Straightforward induction.

Case C-StaticType. Immediate by (T-ANNO).

Case C-FReduce. Immediate by (T-ANNO) and induction.

Case C-AnnoReduce. Immediate by (T-ANNO) and transitivity of <:.

□

Proof for Theorem 2

Proof. Since e is well-typed, by (T-INVK) and (T-ANNO) we know that

$$I <: J, \text{ and } \texttt{mbody}(m, J, J) \text{ is defined}$$

By (T-INTF), $\texttt{mbody}(m, I, J)$ is also defined, and the type checker ensures the expected number of arguments.

On the other hand, since $I <: J$, by the definition of `findOrigin`,

$$\texttt{findOrigin}(m, I, J) \subseteq \texttt{findOrigin}(m, I, I)$$

By (T-NEW), $\texttt{canOverride}(I) = \mathsf{True}$. By the definition of `canOverride`, any $J_0 \in \texttt{findOrigin}(m, I, I)$ satisfies that $\texttt{findOverride}(m, I, J_0)$ contains only one interface, in which the m that overrides J_0 is a concrete method. Therefore $\texttt{mbody}(m, I, J)$ also provides a concrete method, which finishes the proof. □