Appendix to
CMOD: Modular Information Hiding and Type-Safe Linking for C*

Saurabh Srivastava, Michael Hicks, Jeffrey S. Foster
{saurabhs,mwh,jfoster}@cs.umd.edu

1 Introduction

This brief note is an appendix to *CMod: Modular Information Hiding and Type-Safe Linking for C* [2]. It consists of the proof of soundness for the formal language presented in that paper.

2 Soundness

In this section we show that our rules from Figure 2 are sound for MTAL_0, assuming no type abstraction or type naming is present.

We begin by stating some lemmas about MTAL_0 (Figure 4).

Lemma 2.1 (Preservation) *If* $\vdash O_1 \text{ link } O_2 \rightsquigarrow O$ *then* $\vdash O$

Lemma 2.2 (Associativity of link) *If* $\vdash (O_1 \text{ link } O_2) \text{ link } O_3 \rightsquigarrow O$ *then* $\vdash O_1 \text{ link } (O_2 \text{ link } O_3) \rightsquigarrow O$.

Lemma 2.3 (Commutativity of link) *If* $\vdash O_1 \text{ link } O_2 \rightsquigarrow O$ *then* $\vdash O_2 \text{ link } O_1 \rightsquigarrow O$.

Lemma 2.4 *If* $\forall i, j, 1 \leq i, j \leq n, i \neq j \ . \ \vdash O_i \text{ link } O_j \rightsquigarrow O_{ij}$ *and if* π *is any permutation of* $\{1 \ldots n\}$ *then*

$$\vdash O_{\pi(1)} \text{ link } O_{\pi(2)} \text{ link } \ldots \text{ link } O_{\pi(n)} \rightsquigarrow O_{1\ldots n}$$

with $\vdash O_{1\ldots n}$.

We start by observing a property induced by Rule 4.

Lemma 2.5 *If* $\Delta; \mathcal{F} \vdash \mathcal{R}_4(f, \Delta_f)$ *and* $\Delta_f; \mathcal{F} \vdash f \rightsquigarrow \mathcal{A}; \mathcal{I}$ *then* $\Delta; \mathcal{F} \vdash f \rightsquigarrow \mathcal{A}; \mathcal{I}$.

Proof By observation of the rules in Figures 7 and 9. The induction holds trivially for all operational sematics rules except for [IFDEF+] and [IFDEF-]. However, any macros used in those rules are included in $\mathcal{A}^{\mathcal{U}}$, and thus must be either defined in both Δ and Δ_f, or undefined in both, by $\Delta; \mathcal{F} \vdash \mathcal{R}_4(f, \Delta_f)$. . Hence the hypothesis holds for [IFDEF+] and [IFDEF-] as well. □

Next we describe a basic property of [COMPILE] from Figure 3.

Lemma 2.6 *If two fragments have the same preprocessed output then their compiled objects are the same. More formally, if* $\Delta; \mathcal{F} \vdash f_1 \rightsquigarrow \mathcal{A}; \mathcal{I}$ *and* $\Delta; \mathcal{F} \vdash f_2 \rightsquigarrow \mathcal{A}; \mathcal{I}$ *then* $\Delta; \mathcal{F} \vdash f_1 \xrightarrow{comp} O$ *iff* $\Delta; \mathcal{F} \vdash f_2 \xrightarrow{comp} O$

Proof By inspection of rule [COMPILE]. □

*This research was supported in part by NSF CCF-0430118. University of Maryland, Computer Science Department Technical Report CS-TR-4874, UMIACS Tecnical Report UMIACS-TR-2006-13, July 2007.

$$
\begin{array}{rrcl}
\text{program} & \mathcal{P} & ::= & \cdot \mid f \circ \mathcal{P} \\
\text{fragment} & f & := & \cdot \mid s, f \\
\text{trace} & \tilde{f} & := & \cdot \mid \tilde{s}, \tilde{f} \\
\text{statements} & s & ::= & c \mid d \\
\text{preproc. commands} & c & ::= & \mathsf{import}\ h \mid \mathsf{inline}\ h \mid \mathsf{end}\ h \mid \mathsf{def}\ m \mid \mathsf{undef}\ m \\
 & & \mid & \mathsf{ifdef}\ m\ \mathsf{then}\ f\ \mathsf{else}\ f \\
\text{definitions} & d & ::= & \mathsf{let}\ g : \tau = e \mid \mathsf{extern}\ g : \tau \\
 & & \mid & \mathsf{lettype}\ t = \tau \mid \mathsf{type}\ t \\
\text{trace statements} & \tilde{s} & ::= & \tilde{c} \mid d \\
\text{preproc. commands} & \tilde{c} & ::= & \mathsf{import}\ h \mid \mathsf{end}\ h \mid \mathsf{def}\ m \mid \mathsf{undef}\ m \mid \mathsf{ifdef}\ m^{+} \mid \mathsf{ifdef}\ m^{-} \\
\text{terms} & e & ::= & n \mid \lambda y : \tau.\ e \mid e\ e \mid y \mid g \\
\text{types} & \tau & ::= & t \mid \mathsf{int} \mid \tau \to \tau
\end{array}
$$

$m \in$ macro names $\quad g \in$ global var. names $\quad y \in$ local var. names
$h \in$ file names $\quad t \in$ type names

Figure 1: Source Language

One key property the compiler gives us is that, by themselves, each compiled object file is well-formed (in isolation) according to the rules in Figure 4.

Lemma 2.7 (Well-formed compiled objects) *If* $\Delta; \mathcal{F} \vdash f \xrightarrow{comp} [\Psi_I \Rightarrow H : \Psi_E]$ *then* $\vdash [\Psi_I \Rightarrow H : \Psi_E]$

Proof Because [Compile] holds preprocessing f produces an accumulator $(N, H, T, E, I, \mathcal{C}, \mathcal{U}, Z)$. To show that [MTAL$_0$-WF-Obj] holds, we need to identify a heap typing Ψ_A that satisfies the rule. We chose $\Psi_A = N|_{\neg \mathrm{dom}(\Psi_I)}$ from the result accumulator, where $\neg \mathrm{dom}(\Psi_I)$ is any symbol not in the domain of Ψ_I. We now can show that each of the premises of [MTAL$_0$-WF-Obj] hold:

1. $\vdash \Psi_I$. By [Compile] we have $\Psi_I = N|_{(I-E)}$, and by definition there are no duplicate elements in the domain of Ψ_I.

2. $\vdash \Psi_A \leq \Psi_E$. By [Compile] we have $\Psi_E = N|_E$ and $\Psi_I = N|_{(I-E)}$. By construction we have $\Psi_A = N|_{\neg \mathrm{dom}(\Psi_I)}$. But then any symbol in $\mathrm{dom}(\Psi_E)$ must be in $\mathrm{dom}(\Psi_A)$. Furthermore, we have $\vdash \Psi_A$ because by definition there are no duplicate elements in the domain of Ψ_A.

3. $\Psi_I \cup \Psi_A \vdash H : \Psi_A$. By [Compile] we have $N \vdash H$, and then by [WF-Heap] we have $N \vdash e : N(g)$ where $H(g) = e$. The symbol g maps to a unique value in H because of the [WF-Heap]. Additionally, our preprocessor also ensures that symbols are not multiply defined (Rule [Let] in Figure 9). Further, since $\mathrm{dom}(\Psi_I) \cup \mathrm{dom}(\Psi_A) = \mathrm{dom}(N)$ by construction, and since both are projections of N onto smaller domains, we have $\Psi_I \cup \Psi_A = N$. Thus for every $g \in \mathrm{dom}(N)$, we have $\Psi_I \cup \Psi_A \vdash e : N(g)$. Then since $\mathrm{dom}(\Psi_A) \subseteq \mathrm{dom}(N)$, we have $\Psi_I \cup \Psi_A \vdash e : \Psi_A(g)$, which is the same as $\Psi_I \cup \Psi_A \vdash H : \Psi_A$.

4. $\mathrm{dom}(\Psi_I) \cap \mathrm{dom}(\Psi_A) = \emptyset$. This holds trivially, because by [Compile] we have $\Psi_I = N|_{(I-E)}$, and our choice of Ψ_A contains nothing in I in its domain.

□

Now we can prove type-safe linking. Our proof strategy is to show that order-independent fragments have an interpretation consistent with their preprocessing in isolation. We will then use this result to show that if one file imports a symbol and one file exports a symbol, then the Cmod rules force the types to match. Finally, we will show that as a consequence, Cmod enforces type-safe linking.

In this proof, we will use $\mathcal{A}_1 \cup \mathcal{A}_2$ to denote the component-wise union of the two accumulators (which translates to concatenation for any mappings). We also overload the sequencing operator to chain fragments together, so that we may write f_1, f_2 to mean the statements in f_1 followed by the statements in f_2.

We begin by describing the behavior of preprocessing a sequence of fragments:

$$\text{[Sym-Decl]}\quad \frac{h \in \mathcal{I} \qquad \Delta; \mathcal{F} \vdash \mathcal{F}(h) \rightsquigarrow \mathcal{A}; \mathcal{I} \qquad g \in dom(\mathcal{A}^N)}{\Delta; \mathcal{F} \vdash g \overset{\text{decl}}{\longleftarrow} \mathcal{I}}$$

$$\text{[Rule 1]}\quad \frac{\Delta; \mathcal{F} \vdash f_1 \rightsquigarrow \mathcal{A}_1; \mathcal{I}_1 \qquad \Delta; \mathcal{F} \vdash f_2 \rightsquigarrow \mathcal{A}_2; \mathcal{I}_2 \qquad N = \left(\mathcal{A}_1^I \cap \mathcal{A}_2^E\right) \cup \left(\mathcal{A}_1^E \cap \mathcal{A}_2^I\right) \qquad \forall g \in N \;.\; \Delta; \mathcal{F} \vdash g \overset{\text{decl}}{\longleftarrow} \mathcal{I}_1 \cap \mathcal{I}_2}{\Delta; \mathcal{F} \vdash \mathcal{R}_1(f_1, f_2)}$$

(a) Rule 1: Shared Headers

$$\text{[Named-Types-Ok]}\quad \frac{\begin{array}{c}(t \mapsto \tau^\circ) \in T_1 \implies t \notin \text{dom}\,(T_2) \\ (t \mapsto \tau^\circ) \in T_2 \implies t \notin \text{dom}\,(T_1) \\ \left(T_1(t) = \tau_1^{h_1}\right) \wedge \left(T_2(t) = \tau_2^{h_2}\right) \implies h_1 = h_2\end{array}}{\vdash_\tau T_1, T_2}$$

$$\text{[Rule 2]}\quad \frac{\Delta; \mathcal{F} \vdash f_1 \rightsquigarrow \mathcal{A}_1; \mathcal{I}_1 \qquad \Delta; \mathcal{F} \vdash f_2 \rightsquigarrow \mathcal{A}_2; \mathcal{I}_2 \qquad \vdash_\tau \mathcal{A}_1^T, \mathcal{A}_2^T}{\Delta; \mathcal{F} \vdash \mathcal{R}_2(f_1, f_2)}$$

(b) Rule 2: Type Ownership

$$\text{[Trace-Indep]}\quad \frac{\left\langle \mathcal{A}_\emptyset; \tilde{f}_1 \right\rangle \longrightarrow^* \langle \mathcal{A}_1; \cdot \rangle \qquad \left\langle \mathcal{A}_\emptyset; \tilde{f}_2 \right\rangle \longrightarrow^* \langle \mathcal{A}_2; \cdot \rangle \qquad \begin{array}{c}(m, h') \in \mathcal{A}_1^{\mathcal{C}} \wedge (m, h'') \in \mathcal{A}_2^{\mathcal{U}} \Rightarrow h' = h'' \\ (m, h') \in \mathcal{A}_1^{\mathcal{U}} \wedge (m, h'') \in \mathcal{A}_2^{\mathcal{C}} \Rightarrow h' = h''\end{array}}{\tilde{f}_1 \otimes \tilde{f}_2}$$

$$\text{[Partial-Indep]}\quad \frac{\mathcal{F} \vdash \langle \cdot; \cdot; \Delta; f \rangle \longrightarrow^* \left\langle \tilde{f}_1, \mathsf{import}\ h; \mathcal{I}_1; \Delta_1; f_1 \right\rangle \qquad \mathcal{F} \vdash \langle \cdot; \cdot; \Delta; f \rangle \longrightarrow^* \left\langle \tilde{f}_1, \tilde{f}_2, \mathsf{end}\ h; \mathcal{I}_2; \Delta_2; f_1 \right\rangle \qquad \tilde{f}_1 \otimes \tilde{f}_2}{\Delta; \mathcal{F} \vdash f \otimes h}$$

$$\text{[Rule 3]}\quad \frac{\Delta; \mathcal{F} \vdash f \rightsquigarrow \mathcal{A}; \mathcal{I} \qquad \forall h \in \mathcal{I} \;.\; \Delta; \mathcal{F} \vdash f \otimes h}{\Delta; \mathcal{F} \vdash \mathcal{R}_3(f)}$$

(c) Rule 3: Vertical Independence

$$\text{[Rule 4]}\quad \frac{\Delta_f; \mathcal{F} \vdash f \rightsquigarrow \mathcal{A}; \mathcal{I} \qquad ((\Delta - \Delta_f) \cup (\Delta_f - \Delta)) \cap \mathcal{A}^{\mathcal{U}} = \emptyset}{\Delta; \mathcal{F} \vdash \mathcal{R}_4(f, \Delta_f)}$$

(d) Rule 4: Environment Compatibility

$$\text{[All]}\quad \frac{\begin{array}{c}\forall f_1, f_2 \in \mathcal{P} \;.\; f_1 \neq f_2 \;.\; \Delta; \mathcal{F} \vdash \mathcal{R}_1(f_1, f_2) \\ \forall f_1, f_2 \in \mathcal{P} \;.\; f_1 \neq f_2 \;.\; \Delta; \mathcal{F} \vdash \mathcal{R}_2(f_1, f_2) \\ \forall f \in \mathcal{P} \;.\; \Delta; \mathcal{F} \vdash \mathcal{R}_3(f) \\ \forall f \in \mathcal{P} \;.\; \Delta; \mathcal{F} \vdash \mathcal{R}_4(f, \mathcal{E}(f))\end{array}}{\Delta; \mathcal{E}; \mathcal{F} \vdash \mathcal{R}(\mathcal{P})}$$

(e) Rules 1–3 combined

Figure 2: Cmod Rules

$$\text{[WF-Map]}\quad \frac{g_i = g_j \Rightarrow \tau_i = \tau_j}{\vdash g_1 \mapsto \tau_1, \ldots, g_p \mapsto \tau_p}$$

$$\text{[WF-Heap]}\quad \frac{\forall i, j \in [1..p]\ .\ g_i = g_j \Rightarrow i = j \qquad \forall i \in [1..p]\ .\ N \vdash e_i : N(g_i)}{N \vdash g_1 \mapsto e_1, \ldots, g_p \mapsto e_p}$$

$$\text{[Compile]}\quad \frac{\Delta; \mathcal{F} \vdash f \rightsquigarrow (N, H, T, E, I, \mathcal{C}, \mathcal{U}, Z); \mathcal{I} \qquad \vdash N \qquad N \vdash H \qquad \Psi_E = N|_E \qquad \Psi_I = N|_{(I-E)}}{\Delta; \mathcal{F} \vdash f \xrightarrow{\text{comp}} [\Psi_I \Rightarrow H : \Psi_E]}$$

$$\text{[Link]}\quad \frac{\text{dom}\,(H_1) \cap \text{dom}\,(H_2) = \emptyset}{\Delta; \mathcal{F} \vdash [\Psi_{I1} \Rightarrow H_1 : \Psi_{E1}] \circ [\Psi_{I2} \Rightarrow H_2 : \Psi_{E2}] \xrightarrow{\text{comp}} [(\Psi_{I1} \cup \Psi_{I2}) \setminus (\Psi_{E1} \cup \Psi_{E2}) \Rightarrow H_1 \cup H_2 : \Psi_{E1} \cup \Psi_{E2}]}$$

Figure 3: Compiler and Linker Rules

Lemma 2.8 (Preprocessing chains) *If* $\mathcal{F} \vdash \langle \cdot; \cdot; \Delta; f_1 \rangle \longrightarrow^* \left\langle \tilde{f}_1; \mathcal{I}_1; \Delta_1; f_1' \right\rangle$ *then* $\mathcal{F} \vdash \langle \cdot; \cdot; \Delta_1; f_2 \rangle \longrightarrow^* \left\langle \tilde{f}_2; \mathcal{I}_2; \Delta_2; \cdot \right\rangle$ *if and only if* $\mathcal{F} \vdash \langle \cdot; \cdot; \Delta; (f_1, f_2) \rangle \longrightarrow^* \left\langle (\tilde{f}_1, \tilde{f}_2); \mathcal{I}_1 \cup \mathcal{I}_2; \Delta_2; \cdot \right\rangle$

We use Lemma 2.8 without comment in the remainder of the proof.

Next we state a trivial lemma, that preprocessing does not change any macro definitions that are not marked in the accumulator as changed.

Lemma 2.9 *Suppose that* $\mathcal{F} \vdash \langle \cdot; \cdot; \Delta; f \rangle \longrightarrow^* \left\langle \tilde{f}; \mathcal{I}; \Delta'; f' \right\rangle$ *and* $\left\langle \cdot; \tilde{f} \right\rangle \longrightarrow^* \langle \mathcal{A}; \cdot \rangle$ *Then* $\Delta'(m) = \Delta(m)$ *for all* $m \notin \mathcal{A}^{\mathcal{C}}$.

3 Consistent Interpretation

3.1 Proof Strategy

We first discuss our proof strategy by means of an example. The detailed proofs are presented in section 3.2.

Example: Figures 5 and 6 show how the expansion of a header (h_3 in this case) differs when expanded within a fragment and when expanded outside of it.

Once we have proved auxillary lemmas, we can come back to the examples in the figures and notice that the transitive closure under application of the transfer function to the fragment expansions the left (and right) yield fragment expansions in which h_3 is expanded in exactly the way it would be outside of a fragment. We will use the closure of the transfer function to relate two fragments and because the transfer function is accumulator preserving, the accumulator generated by the preprocessing of the header in isolation will be contained in the accumulator generated by preprocessing the fragment in which the header appears.

3.2 Consistent Interpretation using the transfer function

The modified operational semantics for the processor keep track of statements as they are encountered in preprocessing which will be required for defining the transfer function later. The operation of the preprocessor is split into (a) that of producing a *trace* from the given CPP code (Figure 7) and (b) that of producing the preprocessed output from the trace (Figure 9). The preprocessing semantics make use of the function *first-end*$(\cdot)$ defined in Figure 8. Intuitively, *first-end*$(\cdot)$, locates the header name which textually contains

$$[\text{WF-INT}]\ \frac{i \neq j \Rightarrow g_i \neq g_j}{\vdash g_1 \mapsto \tau_1, \ldots, g_p \mapsto \tau_p}$$

$$[\text{INT-SUB}]\ \frac{p \geq q \qquad \vdash g_1 \mapsto \tau_1, \ldots, g_p \mapsto \tau_p}{\vdash g_1 \mapsto \tau_1, \ldots, g_p \mapsto \tau_p \leq g_1 \mapsto \tau_1, \ldots, g_q \mapsto \tau_q}$$

$$[\text{MTAL}_0\text{-WF-OBJ}]\ \frac{\vdash \Psi_I \qquad \vdash \Psi_A \leq \Psi_E \qquad \Psi_I \cup \Psi_A \vdash H : \Psi_A \qquad \text{dom}(\Psi_I) \cap \text{dom}(\Psi_A) = \emptyset}{\vdash [\Psi_I \Rightarrow H : \Psi_E]}$$

$$[\text{MTAL}_0\text{-COMPAT}]\ \frac{\forall g \in \text{dom}(\Psi_1) \cap \text{dom}(\Psi_2)\ .\ \Psi_1(g) = \Psi_2(g)}{\vdash \Psi_1 \sim \Psi_2}$$

$$[\text{MTAL}_0\text{-LC}]\ \frac{\vdash \Psi_{I1} \sim \Psi_{I2} \qquad \vdash \Psi_{I1} \sim \Psi_{E2} \qquad \vdash \Psi_{I2} \sim \Psi_{E1} \qquad \text{dom}(\Psi_{E1}) \cap \text{dom}(\Psi_{E2}) = \emptyset}{\vdash [\Psi_{I1} \Rightarrow H_1 : \Psi_{E1}] \overset{\text{lc}}{\leftrightarrow} [\Psi_{I2} \Rightarrow H_2 : \Psi_{E2}]}$$

$$[\text{MTAL}_0\text{-LINK}]\ \frac{\vdash [\Psi_{I1} \Rightarrow H_1 : \Psi_{E1}] \qquad \vdash [\Psi_{I2} \Rightarrow H_2 : \Psi_{E2}] \qquad \vdash [\Psi_{I1} \Rightarrow H_1 : \Psi_{E1}] \overset{\text{lc}}{\leftrightarrow} [\Psi_{I2} \Rightarrow H_2 : \Psi_{E2}] \qquad \text{dom}(H_1) \cap \text{dom}(H_2) = \emptyset}{\vdash [\Psi_{I1} \Rightarrow H_1 : \Psi_{E1}]\ \text{link}\ [\Psi_{I2} \Rightarrow H_2 : \Psi_{E2}] \rightsquigarrow [(\Psi_{I1} \cup \Psi_{I2}) \setminus (\Psi_{E1} \cup \Psi_{E2}) \Rightarrow H_1 \cup H_2 : \Psi_{E1} \cup \Psi_{E2}]}$$

Figure 4: MTAL_0 [1]

the statement on whose evaluation the function is called. We do not use this intuition in the formal setting but it justifies our use of the term *owner* for the header name returned by the function.

Definition 3.1 (Well-formed traces) *If* $\forall h_i.\mathsf{import}\ h_i \in \tilde{f}$

1. **Expansion:** *There is a unique occurrence each of* $\mathsf{import}\ h_i$ *and* $\mathsf{end}\ h_i$ *both of which come in order.*
2. **Nesting:** $\forall h_j \neq h_i$, *if* $\tilde{f} = (\tilde{f}_s, \mathsf{import}\ h_i, \tilde{f}_i, \mathsf{end}\ h_i, \tilde{f}_e)$, *then* $\mathsf{import}\ h_j \in \tilde{f}_i \iff \mathsf{end}\ h_j \in \tilde{f}_i$
3. **Sequencing:** *The (possibly multiple) occurrences of* $\mathsf{nullimport}\ h_i$ *are strictly after the unique* $\mathsf{end}\ h_i$.

We call a trace, $\tilde{f}$, well formed *if it satisfies the first two properties and* strictly well formed *if it satisfies all three.*

Lemma 3.2 *If* $\mathcal{F} \vdash \langle \cdot; \cdot; \Delta; f \rangle \longrightarrow^* \left\langle \tilde{f}; \mathcal{I}'; \Delta'; \cdot \right\rangle$ *then* $\tilde{f}$ *is strictly well-formed.*

Proof Trivial by examination of Figure 7. □

We use the notation $\left[\tilde{f}\right]_h$ to represent $\mathsf{import}\ h, \tilde{f}, \mathsf{end}\ h$ and by the previous observation on the structure (specifically, the nesting property) of traces generated from fragments this notation would be unambiguous.

The transfer function $\upharpoonright_h$ with respect to h is defined in Figure 10. Intuitively, $\tilde{f}_1 \upharpoonright_h \tilde{f}_2$ relates two traces such that the pair $\left(\left[\tilde{f}_{h'}\right]_{h'}, \mathsf{nullimport}\ h'\right)$ in $\tilde{f}_1$ is replaced with $\left(\mathsf{nullimport}\ h', \left[\tilde{f}_{h'}\right]_{h'}\right)$ in $\tilde{f}_2$. Because we are reducing with respect to h, $\mathsf{nullimport}\ h'$ is constrained to lie in $\left[\tilde{f}_h\right]_h \in \tilde{f}_1$ and obviously, other statements irrelevant to the current reduction are allowed to appear in between the expansion and the $\mathsf{nullimport}$. But we

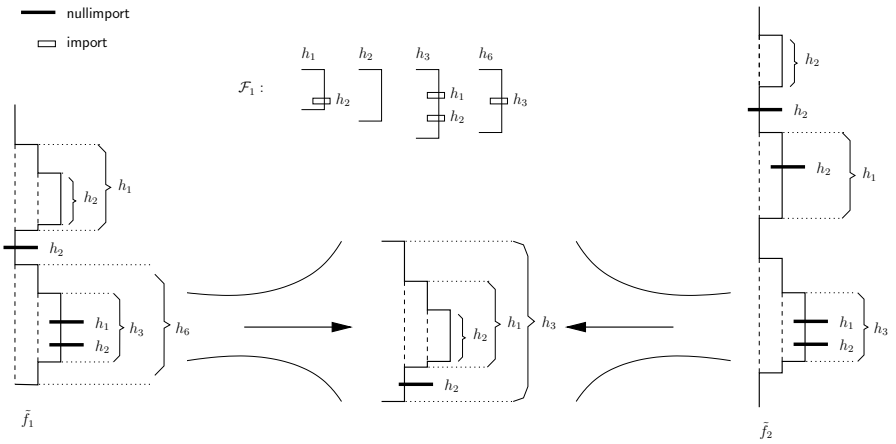

Figure 5: Expanding a header within fragments ($\tilde{f}_1$ and $\tilde{f}_2$ in the figure) might follow different steps than that used to expand the header in isolation (in the middle). Repeated application of $\restriction_{h_3}$ reduces the expansion of either $\tilde{f}_1$ or $\tilde{f}_2$ to one in which h_3 is expanded in exactly the same way as in isolation.

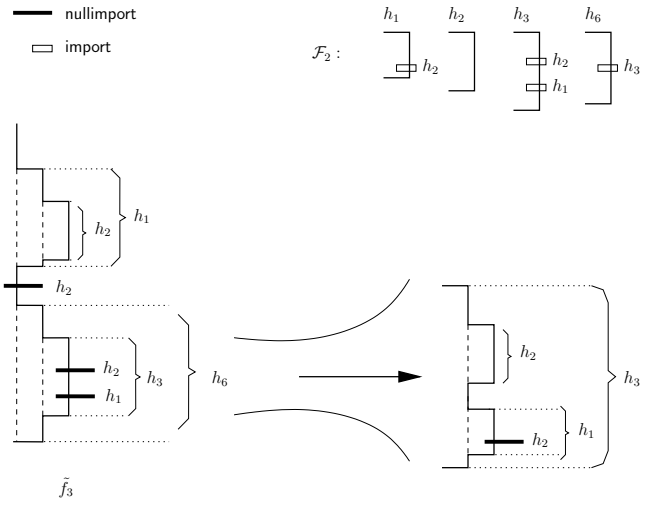

Figure 6: Illustrating the reduction for another filesystem $\mathcal{F}_2$.

$$
\begin{array}{rrcl}
\text{symbols} & N & ::= & \cdot \mid g \to \tau,\ N \\
\text{heap} & H & ::= & \cdot \mid g \to e,\ H \\
\text{named types} & T & ::= & \cdot \mid t \to \tau^h,\ T \mid t \to \tau^\circ,\ T
\end{array}
$$

$$
\begin{array}{rrcl@{\qquad}rrcl}
\text{exports} & E & \in & 2^g & \text{imports} & I & \in & 2^g \\
\text{macro changes} & \mathcal{C} & \in & 2^m \times 2^h & & & & \\
\text{macro uses} & \mathcal{U} & \in & 2^m \times 2^h & \text{type decls} & Z & \in & 2^t \\
\text{includes} & \mathcal{I} & \in & 2^h & \text{defines} & \Delta & \in & 2^m \\
\text{accumulator} & \mathcal{A} & = & (N, H, T, E, I, \mathcal{C}, \mathcal{U}, Z) & & & & \\
\text{file system} & \mathcal{F} & : & h \to f & & & &
\end{array}
$$

[Import]
$$\frac{h \notin \mathcal{I} \qquad \mathcal{I}' = \mathcal{I} \leftarrow^+ h \qquad \tilde{f}' = \tilde{f}, \mathsf{import}\ h}{\mathcal{F} \vdash \left\langle \tilde{f}; \mathcal{I}; \Delta; \mathsf{import}\ h, f \right\rangle \longrightarrow \left\langle \tilde{f}'; \mathcal{I}'; \Delta; \mathcal{F}(h), \mathsf{end}\ h, f \right\rangle}$$

[Import-Empty]
$$\frac{h \in \mathcal{I} \qquad \mathsf{end}\ h \notin f \qquad \tilde{f}' = \tilde{f}, \mathsf{nullimport}\ h}{\mathcal{F} \vdash \left\langle \tilde{f}; \mathcal{I}; \Delta; \mathsf{import}\ h, f \right\rangle \longrightarrow \left\langle \tilde{f}'; \mathcal{I}; \Delta; f \right\rangle}$$

[Eoh]
$$\frac{\tilde{f}' = \tilde{f}, \mathsf{end}\ h}{\mathcal{F} \vdash \left\langle \tilde{f}; \mathcal{I}; \Delta; \mathsf{end}\ h, f \right\rangle \longrightarrow \left\langle \tilde{f}'; \mathcal{I}; \Delta; f \right\rangle}$$

[Inline]
$$\frac{}{\mathcal{F} \vdash \left\langle \tilde{f}; \mathcal{I}; \Delta; \mathsf{inline}\ h, f \right\rangle \longrightarrow \left\langle \tilde{f}; \mathcal{I}; \Delta; \mathcal{F}(h), f \right\rangle}$$

[Def]
$$\frac{\Delta' = \Delta \cup \{m\} \qquad \tilde{f}' = \tilde{f}, \mathsf{def}\ m}{\mathcal{F} \vdash \left\langle \tilde{f}; \mathcal{I}; \Delta; \mathsf{def}\ m, f \right\rangle \longrightarrow \left\langle \tilde{f}'; \mathcal{I}; \Delta'; f \right\rangle}$$

[Undef]
$$\frac{\Delta' = \Delta \setminus \{m\} \qquad \tilde{f}' = \tilde{f}, \mathsf{undef}\ m}{\mathcal{F} \vdash \left\langle \tilde{f}; \mathcal{I}; \Delta; \mathsf{undef}\ m, f \right\rangle \longrightarrow \left\langle \tilde{f}'; \mathcal{I}; \Delta'; f \right\rangle}$$

[Ifdef+]
$$\frac{m \in \Delta \qquad \tilde{f}' = \tilde{f}, \mathsf{ifdef}\ m^+}{\mathcal{F} \vdash \left\langle \tilde{f}; \mathcal{I}; \Delta; (\mathsf{ifdef}\ m\ \mathsf{then}\ f_+\ \mathsf{else}\ f_-), f \right\rangle \longrightarrow \left\langle \tilde{f}'; \mathcal{I}; \Delta; f_+, f \right\rangle}$$

[Ifdef-]
$$\frac{m \notin \Delta \qquad \tilde{f}' = \tilde{f}, \mathsf{ifdef}\ m^-}{\mathcal{F} \vdash \left\langle \tilde{f}; \mathcal{I}; \Delta; (\mathsf{ifdef}\ m\ \mathsf{then}\ f_+\ \mathsf{else}\ f_-), f \right\rangle \longrightarrow \left\langle \tilde{f}'; \mathcal{I}; \Delta; f_-, f \right\rangle}$$

[Extern]
$$\frac{\tilde{f}' = \tilde{f}, \mathsf{extern}\ g : \tau}{\mathcal{F} \vdash \left\langle \tilde{f}; \mathcal{I}; \Delta; \mathsf{extern}\ g : \tau, f \right\rangle \longrightarrow \left\langle \tilde{f}'; \mathcal{I}; \Delta; f \right\rangle}$$

[Let]
$$\frac{\tilde{f}' = \tilde{f}, \mathsf{let}\ g : \tau = e}{\mathcal{F} \vdash \left\langle \tilde{f}; \mathcal{I}; \Delta; \mathsf{let}\ g : \tau = e, f \right\rangle \longrightarrow \left\langle \tilde{f}'; \mathcal{I}; \Delta; f \right\rangle}$$

[Type-Decl]
$$\frac{\tilde{f}' = \tilde{f}, \mathsf{type}\ t}{\mathcal{F} \vdash \left\langle \tilde{f}; \mathcal{I}; \Delta; \mathsf{type}\ t, f \right\rangle \longrightarrow \left\langle \tilde{f}'; \mathcal{I}; \Delta; f \right\rangle}$$

[Type-Def]
$$\frac{\tilde{f}' = \tilde{f}, \mathsf{lettype}\ t = \tau}{\mathcal{F} \vdash \left\langle \tilde{f}; \mathcal{I}; \Delta; \mathsf{lettype}\ t = \tau, f \right\rangle \longrightarrow \left\langle \tilde{f}'; \mathcal{I}; \Delta; f \right\rangle}$$

Figure 7: Operational Semantics for the Preprocessor.

[*first-end*]
$$\frac{\tilde{f} = \tilde{f}', \mathsf{end}\ h, \tilde{f}'' \qquad \mathsf{import}\ h \notin \tilde{f}' \qquad \mathsf{end}\ h'' \in \tilde{f}' \implies \mathsf{import}\ h'' \in \tilde{f}'}{h = \mathit{first\text{-}end}(\tilde{f})}$$

[*first-end*]
$$\frac{\mathsf{end}\ h \in \tilde{f} \implies \mathsf{import}\ h \in \tilde{f}}{\cdot = \mathit{first\text{-}end}(\tilde{f})}$$

Figure 8: The function $\mathit{first\text{-}end}(\tilde{f})$

$$[\textsc{Import}]\ \frac{}{\left\langle \mathcal{A}; \mathsf{import}\ h, \tilde{f} \right\rangle \longrightarrow \left\langle \mathcal{A}; \tilde{f} \right\rangle} \qquad [\textsc{Import-Empty}]\ \frac{}{\left\langle \mathcal{A}; \mathsf{nullimport}\ h, \tilde{f} \right\rangle \longrightarrow \left\langle \mathcal{A}; \tilde{f} \right\rangle} \qquad [\textsc{Eoh}]\ \frac{}{\left\langle \mathcal{A}; \mathsf{end}\ h, \tilde{f} \right\rangle \longrightarrow \left\langle \mathcal{A}; \tilde{f} \right\rangle}$$

$$[\textsc{Def}]\ \frac{h = \mathit{first\text{-}end}(\tilde{f}) \qquad \mathcal{A}' = \mathcal{A}[\mathcal{C} \leftarrow^{+} (m, h),\ \mathcal{U} \leftarrow^{+} (m, h)]}{\left\langle \mathcal{A}; \mathsf{def}\ m, \tilde{f} \right\rangle \longrightarrow \left\langle \mathcal{A}'; \tilde{f} \right\rangle} \qquad [\textsc{Undef}]\ \frac{h = \mathit{first\text{-}end}(\tilde{f}) \qquad \mathcal{A}' = \mathcal{A}[\mathcal{C} \leftarrow^{+} (m, h),\ \mathcal{U} \leftarrow^{+} (m, h)]}{\left\langle \mathcal{A}; \mathsf{undef}\ m, \tilde{f} \right\rangle \longrightarrow \left\langle \mathcal{A}'; \tilde{f} \right\rangle}$$

$$[\textsc{Ifdef}\pm]\ \frac{h = \mathit{first\text{-}end}(\tilde{f}) \qquad \mathcal{A}' = \mathcal{A}[\mathcal{U} \leftarrow^{+} (m, h)]}{\left\langle \mathcal{A}; \mathsf{ifdef}\ m^{\pm}, \tilde{f} \right\rangle \longrightarrow \left\langle \mathcal{A}'; \tilde{f} \right\rangle}$$

$$[\textsc{Extern}]\ \frac{\mathcal{A}' = \mathcal{A}[N \leftarrow^{+} (g \mapsto \tau)]}{\left\langle \mathcal{A}; \mathsf{extern}\ g : \tau, \tilde{f} \right\rangle \longrightarrow \left\langle \mathcal{A}'; \tilde{f} \right\rangle} \qquad [\textsc{Let}]\ \frac{\begin{array}{l}\mathcal{A}' = \mathcal{A}[H \leftarrow^{+} (g \mapsto e),\ N \leftarrow^{+} (g \mapsto \tau),\\ E \leftarrow^{+} g,\ D \leftarrow^{+} g,\ I \leftarrow^{+} \mathrm{fg}\,(e)] \qquad g \notin \mathcal{A}^{H}\end{array}}{\left\langle \mathcal{A}; \mathsf{let}\ g : \tau = e, \tilde{f} \right\rangle \longrightarrow \left\langle \mathcal{A}'; \tilde{f} \right\rangle}$$

$$[\textsc{Type-Decl}]\ \frac{\mathcal{A}' = \mathcal{A}[Z \leftarrow^{+} t]}{\left\langle \mathcal{A}; \mathsf{type}\ t, \tilde{f} \right\rangle \longrightarrow \left\langle \mathcal{A}'; \tilde{f} \right\rangle} \qquad [\textsc{Type-Def}]\ \frac{h = \mathit{first\text{-}end}(\tilde{f}) \qquad \mathcal{A}' = \mathcal{A}[T \leftarrow^{+} (t \mapsto \tau^{h})] \qquad t \notin \mathcal{A}^{T}}{\left\langle \mathcal{A}; \mathsf{lettype}\ t = \tau, \tilde{f} \right\rangle \longrightarrow \left\langle \mathcal{A}'; \tilde{f} \right\rangle}$$

Figure 9: Semantics of construction the accumulator, $\mathcal{A}$, from a trace, $\tilde{f}$: $\left\langle \mathcal{A}_{\emptyset}; \tilde{f} \right\rangle \longrightarrow^{*} \langle \mathcal{A}; \cdot \rangle$

$$[\textsc{Transfer Function}]\ \frac{\mathsf{nullimport}\ h'' \in \tilde{f}_3 \implies \mathsf{import}\ h'' \in \tilde{f}_3}{\tilde{f}_1 \left[\tilde{f}'\right]_{h'} \tilde{f}_2 \left[\tilde{f}_3, \mathsf{nullimport}\ h', \tilde{f}_4\right]_{h} \tilde{f}_5 \quad \wr_h \quad \tilde{f}_1, \mathsf{nullimport}\ h', \tilde{f}_2 \left[\tilde{f}_3 \left[\tilde{f}'\right]_{h'} \tilde{f}_4\right]_{h} \tilde{f}_5}$$

Figure 10: The $\wr_h$ transfer function on traces.

need to constrain the order in which such "pair swaps" occur in order to actually end up with a subtrace that corresponds to the expansion of h that we desire. The precondition that constrains the order of replacement is that in the subtrace ahead of $\mathsf{nullimport}\ h'$ there should be no other unmatched header. Trivially, $\wr_h$ maintains well-formedness (but not strict well-formedness). From this point onwards we use this observation without comment.

As no statement is added or deleted by the transfer function in $\tilde{f}_1 \wr_h \tilde{f}_2$, we can therefore talk about the owner of a statement s under both $\tilde{f}_1$ and $\tilde{f}_2$. We can now show the following lemma:

Lemma 3.3 ($\wr_h$ is ownership preserving) *Let $h_{s,1}$ be the owner of s returned by first-end$(\cdot)$ in $\tilde{f}_1$ and let $h_{s,2}$ be the owner in $\tilde{f}_2$. Then $\tilde{f}_1 \wr_h \tilde{f}_2 \implies h_{s,1} = h_{s,2}$.*

Proof In [Transfer Function] we need to consider cases for statements, s, in each of $\tilde{f}_1, \tilde{f}', \tilde{f}_2, \tilde{f}_3, \tilde{f}_4$ and $\tilde{f}_5$. But what makes the task trivial is the observation that if the structure is of the form $\tilde{f}_a \left[\tilde{f}_x\right]_x \tilde{f}_b$ and s is in $\tilde{f}_a$, then *first-end*$(\cdot)$ can never return an h for which the $\mathsf{end}\ h$ lies in $\tilde{f}_x$ without violating nesting. And therefore a trivial examination of the cases reveals that ownership is maintained in all subtraces. □

Lemma 3.4 ($\wr_h$ is $\mathcal{A}$ preserving) *If*

- *the transfer function relates two preprocessings: $\tilde{f}_1 \wr_h \tilde{f}_2$*

- *the traces yield accumulators as:* $\left\langle \mathcal{A}_\emptyset; \tilde{f}_1 \right\rangle \longrightarrow^* \langle \mathcal{A}_1; \cdot \rangle$ *and* $\left\langle \mathcal{A}_\emptyset; \tilde{f}_2 \right\rangle \longrightarrow^* \langle \mathcal{A}_2; \cdot \rangle$

then $\mathcal{A}_1 = \mathcal{A}_2$.

Proof Notice that since the transfer function does not add or delete any statements, both accumulators are constructed over exactly the same *set* of statements, albeit in differing order. Then, the only way the accumulators corresponding to the two traces can be different is if *first-end*$(\cdot)$ differs for some statement in the evaluation, and this cannot happen due to Lemma 3.3. □

Corollary 3.5 *If* $\left\langle \mathcal{A}_\emptyset; \tilde{f}_1 \right\rangle \longrightarrow^* \langle \mathcal{A}_1; \cdot \rangle$ *and* $\left\langle \mathcal{A}_\emptyset; \tilde{f}_2 \right\rangle \longrightarrow^* \langle \mathcal{A}_2; \cdot \rangle$ *then* $\tilde{f}_1 \upharpoonright_h^* \tilde{f}_2 \implies \mathcal{A}_1 = \mathcal{A}_2$.

Proof Trivial proof by induction on the number of steps in $\upharpoonright_h^*$ and repeated application of Lemma 3.4. □

Lemma 3.6 *If*

- $\tilde{f}_h$ *occurs in* $\tilde{f}$ *(i.e.* $\tilde{f} = \tilde{f}_a \left[\tilde{f}_h\right]_h \tilde{f}_b$*)*
- $\left\langle \mathcal{A}_\emptyset; \tilde{f} \right\rangle \longrightarrow^* \langle \mathcal{A}; \cdot \rangle$ *and* $\left\langle \mathcal{A}_\emptyset; \tilde{f}_h, \mathsf{end}\ h \right\rangle \longrightarrow^* \langle \mathcal{A}_h; \cdot \rangle$

then $\mathcal{A}_h \subseteq \mathcal{A}$.

To help prove the next lemma we would need the notion of the *canonical* expansion trace of a header, h, the *consistency* of two traces and the notion of a *distance* of a trace with respect to a header. We define these below:

Definition 3.7 (Canonical trace for a header, C_h) *If* $\mathcal{F} \vdash \langle \cdot; \cdot; \Delta; \mathcal{F}(h), \mathsf{end}\ h \rangle \longrightarrow^* \left\langle \tilde{f}_h; \mathcal{I}_h; \Delta_h; \mathsf{end}\ h \right\rangle$ *then we call* $\tilde{f}_h$ *the the* canonical *trace, denoted by* C_h*, for a header* h.

Definition 3.8 (Consistency of traces) *We call a trace,* $\tilde{f}_1$*, as being* consistent *with* $\tilde{f}_2$*, iff*

- $\tilde{f}_1 = \tilde{f}_2$*, or*
- $\tilde{f}_1 = (\tilde{f}_{1a}, \mathsf{nullimport}\ h, \tilde{f}_{1b})$ *and* $\tilde{f}_2 = (\tilde{f}_{2a}, \mathsf{import}\ h, \tilde{f}_h, \mathsf{end}\ h, \tilde{f}_{2b})$*, where* $\tilde{f}_{1a}$ *is consistent with* $\tilde{f}_{2a}$ *and* $\tilde{f}_{1b}$ *is consistent with* $\tilde{f}_{2b}$.

Note that the definition of consistency is asymmetric. Also consistency is a transitive relation. In other words, if $\tilde{f}_x$ is consistent with $\tilde{f}_y$ and $\tilde{f}_y$ is consistent with $\tilde{f}_z$ then $\tilde{f}_x$ is consistent with $\tilde{f}_z$. The following is a trivial consequence of the above definition:

Corollary 3.9 *If* $\tilde{f}_1$ *is consistent with* $\tilde{f}_2$ *for a strictly well formed* $\tilde{f}_2$ *and* $\left[\tilde{f}_{h,2}\right]_h \in \tilde{f}_2$ *and* $\exists \tilde{f}_{h,1}$ *consistent with* $\tilde{f}_{h,2}$ *then* $\tilde{f}_1 \left[\mathsf{nullimport}\ h \mapsto \left[\tilde{f}_{h,1}\right]_h\right]$ *is consistent with* $\tilde{f}_2$*, where the substitution is done for the first occurrence.*

[Rule 3] allows us to relate the consistency of two preprocessings as follows:

Lemma 3.10 *If*

- $\mathcal{F} \vdash \langle \cdot; \cdot; \Delta_g; f_1 \rangle \longrightarrow^* \left\langle \tilde{f}_x, \mathsf{import}\ h; \mathcal{I}_x; \Delta_x; f_x \right\rangle \longrightarrow^* \left\langle \tilde{f}_x, \tilde{f}_y, \mathsf{end}\ h; \mathcal{I}_y; \Delta_y; f_y \right\rangle$
- $\mathcal{F} \vdash \langle \cdot; \cdot; \Delta_g; f_2 \rangle \longrightarrow^* \left\langle \tilde{f}_a, \mathsf{import}\ h; \mathcal{I}_a; \Delta_a; f_a \right\rangle \longrightarrow^* \left\langle \tilde{f}_a, \tilde{f}_b, \mathsf{end}\ h; \mathcal{I}_b; \Delta_b; f_b \right\rangle$
- $\mathcal{I}_a \subseteq \mathcal{I}_x$
- $\Delta_g; \mathcal{F} \vdash \mathcal{R}_3(f_1)$ *(which implies* $\tilde{f}_x \otimes \tilde{f}_y$*)*

- $(m, h') \in \mathcal{A}_y^{\mathcal{U}} \implies (m, h'') \notin \mathcal{A}_a^{\mathcal{C}}$

then $\tilde{f}_y$ is consistent with $\tilde{f}_b$.

Proof Our proof strategy is as follows: We imagine different filesystems, $\{\mathcal{F}_1, \mathcal{F}_2 \ldots\}$, from the one that we have at hand. This set of filesystems is not arbitrarily different but match up with the given filesystem, $\mathcal{F}$, as specified later. $\mathcal{F}_t$ is such that it takes exactly t steps to reduce h in the fragment f_1 and these are exactly the first t steps taken under $\mathcal{F}$. It might be the case that the first t steps under $\mathcal{F}$ might leave some headers hanging with just imports having been seen without their corresponding ends. For these we simply add as many ends, in the right order, as required, without counting them in t. Also, $\forall i \;.\; \forall h \in \mathcal{I}_x \;.\; \mathcal{F}(h) = \mathcal{F}_i(h)$.

(3.10) By induction on the number of steps in the expansion of h. The base case, $n = 1$, corresponds to the header being empty and the expansion of h going to end h in one step. Then the lemma holds trivially.

We first make a simple observation which we use in the induction step:

Observation 3.11 *If $\tilde{f}_1$ is consistent with $\tilde{f}_1'$ and $\tilde{f}_2$ is consistent with $\tilde{f}_2'$, then $\tilde{f}_1, \tilde{f}_2$ is consistent with $\tilde{f}_1', \tilde{f}_2'$.*

For the inductive case, assume that the filesystem is such that the header, h, expands in exactly $n = k$ steps. i.e. the filesystem is $\mathcal{F}_k$. We assume that the lemma holds in that case and show that it also holds for another filesystem, $\mathcal{F}_{k+1}$, in which the header expands identically except for one more statement, s in the end. Formally, we have the following induction hypothesis:

$$\mathcal{F}_k \vdash \langle \cdot; \cdot; \Delta_g; f_1 \rangle \longrightarrow^* \left\langle \tilde{f}_x, \mathsf{import}\ h; \mathcal{I}_x; \Delta_x; f_x \right\rangle \longrightarrow^k \left\langle \tilde{f}_x, \tilde{f}_y, \mathsf{end}\ h; \mathcal{I}_y; \Delta_y; f_y \right\rangle$$
$$\mathcal{F}_k \vdash \langle \cdot; \cdot; \Delta_g; f_2 \rangle \longrightarrow^* \left\langle \tilde{f}_a, \mathsf{import}\ h; \mathcal{I}_a; \Delta_a; f_a \right\rangle \longrightarrow^* \left\langle \tilde{f}_a, \tilde{f}_b, \mathsf{end}\ h; \mathcal{I}_b; \Delta_b; f_b \right\rangle$$

And the induction step is:

$$\mathcal{F}_{k+1} \vdash \langle \cdot; \cdot; \Delta_g; f_1 \rangle \longrightarrow^* \left\langle \tilde{f}_x, \mathsf{import}\ h; \mathcal{I}_x; \Delta_x; f_x \right\rangle \longrightarrow^{k+1} \left\langle \tilde{f}_x, \tilde{f}_y, s, \mathsf{end}\ h; \mathcal{I}_y'; \Delta_y'; f_y \right\rangle$$
$$\mathcal{F}_{k+1} \vdash \langle \cdot; \cdot; \Delta_g; f_2 \rangle \longrightarrow^* \left\langle \tilde{f}_a, \mathsf{import}\ h; \mathcal{I}_a; \Delta_a; f_a \right\rangle \longrightarrow^* \left\langle \tilde{f}_a, \tilde{f}_b, \tilde{f}_s, \mathsf{end}\ h; \mathcal{I}_b'; \Delta_b'; f_b \right\rangle$$

The invariants that we maintain during the induction step are the following:

(a) $\tilde{f}_y$ is consistent with $\tilde{f}_b$
(b) $\mathcal{I}_b \subseteq \mathcal{I}_y$
(c) *first-end*$(\cdot)$ is identical in both traces

Notice that we are inducting on the the number of steps taken in the reduction for f_1. Also, the steps in the initial part of the reduction need to be identical. We now consider cases the extra s and show that the lemma holds in each case:

- $s \in \{\mathsf{extern}, \mathsf{let}, \mathsf{type}, \mathsf{lettype}, \mathsf{end}\}$: $n = k + 1$ trivial from $n = k$.
- $s \in \{\mathsf{def}, \mathsf{undef}\}$: We argue for the case of $s = \mathsf{def}\ m$. The argument for undef m is identical. From the induction hypothesis we know that $\tilde{f}_y$ is consistent with $\tilde{f}_b$ and therefore it is still the case that $\tilde{f}_y, \mathsf{def}\ m$ is consistent with $\tilde{f}_b, \mathsf{def}\ m$ for reduction under $\mathcal{F}_{k+1}$. In this case $\tilde{f}_s = s$, $\mathcal{I}_b' = \mathcal{I}_b$ and $\mathcal{I}_y' = \mathcal{I}_y$ and therefore the lemma holds for $n = k + 1$.
- $s = \mathsf{ifdef}\ m^{\pm}$: We again assume that the lemma holds for k reduction steps and we append a ifdef $m^{\pm}$ to the end. We have to ensure that the sign on the statement is the same in both reductions. From the last assumption in the lemma we know that $m \notin \mathcal{A}_a^{\mathcal{C}}$. Also by the fact that $\tilde{f}_x \otimes \tilde{f}_y$ we know that either $m \notin \mathcal{A}_x^{\mathcal{C}}$. m might have been changed in $\tilde{f}_y$, but we know that each statement in $s \in \tilde{f}_x \implies s \in \tilde{f}_b$ and therefore the sign on the macro is the same in both reductions. Therefore, $\tilde{f}_s = s$ for this case and $\mathcal{I}_b' = \mathcal{I}_b$ and $\mathcal{I}_y' = \mathcal{I}_y$ and therefore the lemma holds for $n = k + 1$.

- $s = \mathsf{nullimport}\ h'$: This can happen only if $h' \in \mathcal{I}_y$. Then there are two cases:
  - $h' \in \mathcal{I}_b$: Then in both reductions [IMPORT-EMPTY] is applied and $\tilde{f}_s = s$. Also, $\mathcal{I}'_b = \mathcal{I}_b$ and $\mathcal{I}'_y = \mathcal{I}_y$ and therefore the lemma holds for $n = k + 1$.
  - $h' \notin \mathcal{I}_b$: In this case $\mathsf{nullimport}\ h'$ goes to $\tilde{f}_y$ while the header is expanded in $\tilde{f}_b$. $\tilde{f}_s = \left[\tilde{f}_{h'}\right]_{h'}$ which implies that s is consistent with $\tilde{f}_s$. And therefore by Observation 3.11, $\tilde{f}_y, s$ is consistent with $\tilde{f}_b, \tilde{f}_s$. Let $H = \{h'\} \cup \left\{h'' \,\middle|\, \left[\tilde{f}_{h''}\right]_{h''} \in \tilde{f}_{h'}\right\}$, then $\mathcal{I}'_b = \mathcal{I}_b \cup H$. By the assumptions $\tilde{f}_x \otimes \tilde{f}_y$ and the last assumption in the lemma we know that $\forall h'' \in H\ .\ h'' \in \mathcal{I}_x$ and therefore $\mathcal{I}_b \subseteq \mathcal{I}_y \implies \mathcal{I}'_b \subseteq \mathcal{I}'_y$.
- $s = \mathsf{import}\ h'$: This happens when $h' \notin \mathcal{I}_y$ (and consequently, $\mathcal{I}_b \subseteq \mathcal{I}_y \implies h' \notin \mathcal{I}_b$) then in both cases [IMPORT] is applied. And therefore $\tilde{f}_s = s$ and $\mathcal{I}'_b = \mathcal{I}_b \cup \{h'\}$ and $\mathcal{I}'_y = \mathcal{I}_y \cup \{h'\}$ and therefore the lemma holds for $n = k + 1$. Notice that $h' \notin \mathcal{I}_y \implies h' \notin \mathcal{I}_x$ and therefore this is the first time h' is being expanded and therefore by inducting inside of it we are not making it any more difficult to find subsequent $\mathcal{F}_j$'s ($j > k + 1$) for which the initial reductions would be identical. □

We define the notation $\left\langle h \,\middle|\, \mathsf{nullimport}\ h \in \tilde{f} \right\rangle$ to mean the *ordered set* of header names where the ordering is defined based on left to right appearance of the $\mathsf{nullimport}$ statements in $\tilde{f}$. Let $\tilde{f}$ be a well formed trace and let $\left[\tilde{f}_{h_i}\right]_{h_i}$ be the unique expansion of h_i in $\tilde{f}$. Let us construct a *nullorder* tree from a starting set of *ordered* header names, H, under a given trace, $\tilde{f}$, as follows. We give two definitions, both of which work:

- Make a root and make the elements in H the ordered children of the root. Then apply the following operation repeatedly. For occurrences of h_i that do not already have children, add the ordered set $\left\langle h' \,\middle|\, \mathsf{nullimport}\ h' \in \left[\tilde{f}_{h_i}\right]_{h_i} \right\rangle$ as the ordered children of that node.
- Make a root and make the elements in H the ordered children of the root. Then apply the following operation repeatedly until you reach a fixpoint tree. In the preorder traversal of the current tree, find the occurrence of h_i, which appears before any other h_i in the order traversal and if the node does not already have children, then add the ordered set $\left\langle h' \,\middle|\, \mathsf{nullimport}\ h' \in \left[\tilde{f}_{h_i}\right]_{h_i} \right\rangle$ as the ordered children of that node.

The first definition creates an infinite tree over a finite set of node names and is easy to understand. The second creates a finite tree but is more complicated to evaluate. We now define two functions:

$$\mathsf{close}_{\tilde{f}}(H) = \begin{array}{l}\text{preorder traversal, ignoring duplicates and}\\ \text{the root, of nullorder tree for } H \text{ under } \tilde{f}\end{array} \tag{3.1}$$

$$\mathsf{collapse}_H\left(\tilde{f}\right) = \forall_{h \in H\ ,\ \left[\tilde{f}_h\right]_h \in \tilde{f}}\ .\ \tilde{f}\left[\left[\tilde{f}_h\right]_h \mapsto \mathsf{nullimport}\ h\right] \tag{3.2}$$

The above definition allows us to state the following corollaries. Notice that in Corollary 3.13 below, we are stating that even though it might be the case that $\mathcal{I}_a \not\subseteq \mathcal{I}_x$, but if $\mathcal{I}_a \subseteq \mathcal{I}_x \cup H$, for some H, then a statement can be made about consistency.

Observation 3.12 *If $\tilde{f}, \tilde{f}'$ are traces and H_x is a set such that $\forall h \in H_x\ .\ \left[\tilde{f}_h\right]_h \notin \tilde{f}$ then*
$\mathsf{collapse}_{H \cup H_x}\left(\tilde{f}\right)$ *is consistent with* $\tilde{f}' \implies \mathsf{collapse}_H\left(\tilde{f}\right)$ *is consistent with* $\tilde{f}'$

Corollary 3.13 *If there exist fragments f_1 and f_2 which expand as in Lemma 3.10 and $\mathcal{I}_a \subseteq \mathcal{I}_x \cup H$ for some H and $\Delta; \mathcal{F} \vdash \mathcal{R}_3(f_1)$ and for each $h \in H$ the expansion of h in $\tilde{f}_1$, $\left[\tilde{f}_h\right]_h$, satisfies $\tilde{f}_h \otimes \tilde{f}_y$ then* $\mathsf{collapse}_H\left(\tilde{f}_y\right)$ *is consistent with* $\tilde{f}_b$.

Proof The proof is almost identical to that of Lemma 3.10. The induction step now becomes:

$$\underset{\mathcal{I}_b \subseteq \mathcal{I}_y \cup H}{\mathsf{collapse}_H\Big(\tilde{f}_y\Big) \text{ is consistent with } \tilde{f}_b} \quad \text{should imply} \quad \underset{\mathcal{I}'_b \subseteq \mathcal{I}'_y \cup H}{\mathsf{collapse}_H\Big(\tilde{f}_y, s\Big) \text{ is consistent with } \tilde{f}_b, \tilde{f}_s}$$

The only cases of significance are of $s = \{\mathsf{nullimport}\ h', \mathsf{import}\ h'\}$:

- $s = \mathsf{nullimport}\ h'$: This can happen only if $h' \in \mathcal{I}_y$. Then there are two cases:
  - $h' \in \mathcal{I}_b$: Then in both reductions [IMPORT-EMPTY] is applied and $\tilde{f}_s = s$. Also, $\mathcal{I}'_b = \mathcal{I}_b$ and $\mathcal{I}'_y = \mathcal{I}_y$ and therefore the lemma holds for $n = k + 1$.
  - $h' \notin \mathcal{I}_b$: In this case $\mathsf{nullimport}\ h'$ goes to $\tilde{f}_y$ while the header is expanded in $\tilde{f}_b$. $\tilde{f}_s = \left[\tilde{f}_{h'}\right]_{h'}$ which implies that s is consistent with $\tilde{f}_s$. And therefore by Observation 3.11, $\tilde{f}_y, s$ is consistent with $\tilde{f}_b, \tilde{f}_s$. $\mathcal{I}'_b = \mathcal{I}_b \cup \{h'\}$ but in this case we know that $h' \in \mathcal{I}_y$ and therefore $\mathcal{I}_b \subseteq \mathcal{I}_y \implies \mathcal{I}'_b \subseteq \mathcal{I}'_y$.
- $s = \mathsf{import}\ h'$: This happens when $h' \notin \mathcal{I}_y$
  - $h' \notin H$: $\mathcal{I}_b \subseteq \mathcal{I}_y \cup H \implies h' \notin \mathcal{I}_b$. Then in both reductions [IMPORT] is applied. And therefore $\tilde{f}_s = s$ and $\mathcal{I}'_b = \mathcal{I}_b \cup \{h'\}$ and $\mathcal{I}'_y = \mathcal{I}_y \cup \{h'\}$ and therefore the lemma holds for $n = k + 1$.
  - $h' \in H$: The two cases here are $h' \in \mathcal{I}_b$ and $h' \notin \mathcal{I}_b$. In the case of $h' \in \mathcal{I}_b$, [IMPORT-EMPTY] is applied and $\mathsf{nullimport}\ h'$ is appended to trace $\tilde{f}_b$. But because $h \in H$, it is the case that the expansion appearing in $\tilde{f}_y$ is compressed to a $\mathsf{nullimport}\ h'$ by collapse. And therefore $\mathsf{collapse}_H\Big(\tilde{f}_y, \left[\tilde{f}''\right]_{h'}\Big)$ is consistent with $\tilde{f}_b, \mathsf{nullimport}\ h'$. On the other hand if $h' \notin \mathcal{I}_b$, then [IMPORT] is applied in both preprocessings and the argument reduces as for the case of $h' \notin H$.

  Notice that $h' \notin \mathcal{I}_y \implies h' \notin \mathcal{I}_x$ and therefore this is the first time h' is being expanded and therefore by inducting inside of it we are not making it any more difficult to find subsequent $\mathcal{F}_j$'s ($j > k + 1$) for which the initial reductions would be identical. □

In the following, for an ordered set H and a header name h we define the *restriction*, $H|_{<h}$, to be another ordered set constructed from the elements of H that appear strictly before h in H, with their order preserved.

Definition 3.14 ($\tilde{f}$ is h–nice) *Let $\left[\tilde{f}_h\right]_h \in \tilde{f}$ be the expansion of h in $\tilde{f}$. Then we call $\tilde{f}$, h–nice if:*

3.14a *$\tilde{f}_h$ is consistent with C_h and*

3.14b *Let*

$$H = \left\langle h' \,\middle|\, \mathsf{nullimport}\ h' \in \tilde{f}_h \right\rangle \tag{3.3}$$

$$U^* = \mathsf{close}_{\tilde{f}}(H) \setminus \left\{ h' \,\middle|\, \mathsf{import}\ h' \in \tilde{f}_h \right\} \tag{3.4}$$

$$\forall h' \in U^* \qquad H^*_{h'} = U^*|_{<h'}; \quad \left[\tilde{f}_{h',1}\right]_{h'} \in \tilde{f}; \quad \left[\tilde{f}_{h',2}\right]_{h'} \in C_h \tag{3.5}$$

then $\forall h' \in U^*$. $\mathsf{collapse}_{H^*_{h'}}\Big(\tilde{f}_{h',1}\Big)$ is consistent with $\tilde{f}_{h',2}$

Statement **3.14b** can be understood better by rephrasing as follows: For all unmatched $h_k \in \tilde{f}_h$ it is the case that the collapse of its expansion under H^*_k (which is the closure of the set of unmatched headers before h_k in $\tilde{f}_h$) yields a trace consistent with h_k's expansion in C_h. We start our investigation of the relation of $\wr_h$ with that of traces being h–nice by making a simple observation:

Observation 3.15 *If we discount the headers inside the expansion of h then an application of $\restriction_h$ is order preserving with respect to the nullorder tree.*

Proof If $\tilde{f}_1 \restriction_h \tilde{f}_2$ and let $\left[\tilde{f}_{h,1}\right]_h \in \tilde{f}_1$ and $\left[\tilde{f}_{h,2}\right]_h \in \tilde{f}_2$ and by Eq. 3.3 let us defined H_1 and H_2 using $\tilde{f}_{h,1}$ and $\tilde{f}_{h,2}$, respectively. Also let $X_{h,1}$ and $X_{h,2}$ be the header names as defined by the second term in Eq. 3.4 using $\tilde{f}_{h,1}$ and $\tilde{f}_{h,2}$, respectively. Then the observation is saying that discounting headers in $X_{h,1}$ (respectively, $X_{h,2}$) the nullorder trees constructed using H_1 and H_2 have the same preorder traversals.

To see this let us look at the effect of an application of $\restriction_h$ on the nullorder tree. There are two operations:

(a) A $\mathsf{nullimport}\ h'$ is replaced with expansion outside of h. This operation can easily be seen to be equivalent to replacing a child of the root with its children. And the replaced node is in $X_{h,2}$ and so this operation does not have any effect on the preorder traversal.

(b) A $\mathsf{nullimport}\ h'$ is moved to a location outside of h. This case is a little trickier to visualize, but is much easier said with an example: Consider $\left[\ldots\left[\ldots\left[\tilde{f}_{h'}\right]_{h'}\ldots\right]_{h''}\ldots\right]_{h'''}\ldots\left[\ldots\mathsf{nullimport}\ h'\ldots\right]_h$ and suppose that h'' and h''' are in the nullorder tree. In this case, notice that any $\mathsf{nullimport}$s that are inside $\tilde{f}_{h'}$ are children of both h'' and h'''. Replacing $\left[\tilde{f}_{h'}\right]_{h'}$ with $\mathsf{nullimport}\ h'$ causes those children to not be there anymore. But (!) notice that because $\mathsf{nullimport}\ h'$ is now a child of h'' and h''', those children are now their grandchildren. And again, since h' is in $X_{h,2}$ and hence discounted, we again have that the preorder traversal is preserved.

Notice that these two are the only operations affecting the tree. The specific definition of the tree causes other considerations to be abstracted out. For instance, it might be the case that $\exists x\ .\ \left[\tilde{f}_x\right]_x \in \tilde{f}_{h'}$. The fact that it is moved along with $\left[\tilde{f}_{h'}\right]_{h'}$ does not affect the subtrees rooted at x, because $\tilde{f}_x$ and the $\mathsf{nullimport}$ x's are unchanged. □

Lemma 3.16 *If $\tilde{f}_1 \restriction_h \tilde{f}_2$ then $\tilde{f}_1$ is h–*nice* $\implies$ $\tilde{f}_2$ is h–*nice*.*

Proof We have to prove that both 3.14a and 3.14b from Definition 3.14 hold after the application of $\restriction_h$.

3.14a From [Transfer Function] we know that $\tilde{f}_1$ has to be such that $\left[\tilde{f}_h\right]_h \in \tilde{f}_1$ where $\tilde{f}_h = \tilde{f}_3, \mathsf{nullimport}\ h', \tilde{f}_4$ with the condition that $\mathsf{nullimport}\ h'' \in \tilde{f}_3 \implies \mathsf{import}\ h'' \in \tilde{f}_3$. For h', Eq. (3.5) in Definition 3.14b defines $H^*_{h'}$ and our intention is to show that $H^*_{h'} = \emptyset$. Suppose $\exists h''' \in H^*_{h'}$. Then from the subtraction of imports in Eq. (3.4) we know that $\mathsf{import}\ h''' \notin \tilde{f}_3, \tilde{f}_4$. Since $\mathsf{nullimport}\ h' \in \tilde{f}_h$ it implies that $h' \in H$ in Eq. (3.3). Then from the construction of the nullorder tree and the definition of restriction we know that if h' is a child of the root and there is an h''' before it in U^* then $\exists h''''.\mathsf{nullimport}\ h'''' \in \tilde{f}_3$ such that h'''' is an ancestor of h''' and a left sibling of h'. But by the precondition in [Transfer Function] this implies that $\mathsf{import}\ h'''' \in \tilde{f}_3$. If $\mathsf{import}\ h'''' \in \tilde{f}_3$ then all its children in the nullorder tree have to be such that $\mathsf{nullimport}$s are in $\tilde{f}_h$. And then we identically argue to say that its children have to have their imports in $\tilde{f}_h$ and so on till we arrive at h''' and at a contradiction. Therefore $H^*_{h'} = \emptyset$. This argument proves that h' is the first element in U^* for which $\mathsf{nullimport}\ h' \in \tilde{f}_h \not\Longrightarrow \mathsf{import}\ h' \in \tilde{f}_h$.

Let $\left[\tilde{f}_{h'}\right]_{h'} \in \tilde{f}_1$ be the expansion of h' in $\tilde{f}_1$ and $\left[\tilde{f}'_{h'}\right]_{h'}$ be the expansion of h' in C_h. Then a simple observation of Eq. (3.2) with $H^*_{h'} = \emptyset$ indicates that $\mathsf{collapse}_{H^*_{h'}}\left(\tilde{f}_{h'}\right) \equiv \tilde{f}_{h'}$ (i.e. identity). $\tilde{f}_1$ is h–*nice* requires that $\mathsf{collapse}_{H^*_{h'}}\left(\tilde{f}_{h'}\right)$ be consistent with $\tilde{f}'_{h'}$ and therefore $\tilde{f}_{h'}$ is consistent $\tilde{f}'_{h'}$. Therefore, by [Transfer Function] and Corollary 3.9 we know that $\tilde{f}_3\left[\tilde{f}_{h'}\right]_{h'}\tilde{f}_4$ is consistent with C_h.

3.14b Let h' be the header being manipulated as in the proof of **3.14a**, i.e., let $\left[\tilde{f}_{h'}\right]_{h'} \in \tilde{f}_1$ be the expansion of h' in $\tilde{f}_1$ which is swapped with $\mathsf{nullimport}\ h'$. Let $U^*_{\tilde{f}_1}$ and $U^*_{\tilde{f}_2}$ be the computation of Eq. (3.4) for $\tilde{f}_1$ and $\tilde{f}_2$, respectively. An immediate consequence of Observation 3.15 is that $U^*_{\tilde{f}_1} \supseteq U^*_{\tilde{f}_2}$. This implies that in Eq. (3.5) the universal quantification if over the same set of header names. Therefore we only need to show that $\forall h'' \in U^*_{\tilde{f}_1}$, the statement of **3.14b** holds after the application of $\restriction_h$. The transfer function makes two changes to the trace, one of moving the $\mathsf{nullimport}\ h'$ outside and another of moving $\left[\tilde{f}_{h'}\right]_{h'}$ inside of $\left[\tilde{f}_h\right]_h$. We have to reason about each in turn.

There are two cases to consider with regard to the $\mathsf{nullimport}\ h'$ that is moved outside:

(a) $\exists h'' \,.\, h'' \in U^*_{\tilde{f}_1} \wedge \left[\tilde{f}_{h'}\right]_{h'} \in \left[\tilde{f}_{h''}\right]_{h''}$: The occurrence of the $\left[\tilde{f}_{h'}\right]_{h'}$ in $\tilde{f}_{h''}$ is replaced with a $\mathsf{nullimport}\ h'$ and condition **3.14b** on $\tilde{f}_1$, specifically that $\mathsf{collapse}_{H^*_{h''}}\left(\tilde{f}_{h''}\right)$ is consistent with h''s expansion in C_h, ensures that after the substitution $\tilde{f}_{h''}$ remains consistent with the corresponding expansion in C_h. This observation along with $U^*_{\tilde{f}_1} \supseteq U^*_{\tilde{f}_2}$ guarantees that the substitution with $\mathsf{nullimport}\ h'$ cannot violate consistency.

(b) otherwise: The $\mathsf{nullimport}\ h'$ is placed at the toplevel in $\tilde{f}_1$ or in some $\left[\tilde{f}_{h''}\right]_{h''}$, where $h'' \notin U^*_{\tilde{f}_1}$, and therefore no header expansion of concern is affected.

We now reason about the effects of moving $\left[\tilde{f}_{h'}\right]_{h'}$ inside of $\left[\tilde{f}_h\right]_h$. Because of the just observed effect of the moving the $\mathsf{nullimport}\ h'$ outside, we reason that $\forall h'' \in U^*_{\tilde{f}_1} (\supseteq U^*_{\tilde{f}_2})$ unless the corresponding $H^*_{h''}$ diminishes, $\mathsf{collapse}_{H^*_{h''}}\left(\left[\tilde{f}_{h''}\right]_{h''}\right)$ will remain consistent with the application of $\restriction_h$. Therefore the only case to consider is if for some h'' the corresponding $H^*_{h''}$ diminishes. Let $H^*_{h'',1}, H^*_{h'',2}$ be the values of the sets before and after the application of the transfer function. We show later that $H^*_{h'',1} \setminus H^*_{h'',2} \subseteq H_{\text{new}}$ where $H_{\text{new}} = \left\{ h_x \,\middle|\, \left[\tilde{f}_{h_x}\right]_{h_x} \in \left[\tilde{f}_{h'}\right]_{h'} \right\} \cup \{h'\}$, where we recall that $\left[\tilde{f}_{h'}\right]_{h'}$ is the subtrace being swapped in the application of rule [Transfer Function]. We now show that $H^*_{h'',1} \setminus H^*_{h'',2} \subseteq H_{\text{new}}$ completes the proof of consistency for each h''. Each h'' can fall into one of two categories (depending on the structure of its expansion $\left[\tilde{f}''\right]_{h''}$):

(a) $\left[\tilde{f}_{h'}\right]_{h'} \in \left[\tilde{f}''\right]_{h''} \in \tilde{f}_1$: (This h'' is the unique 'container' of h' in $\tilde{f}_1$) The application of $\restriction_h$ replaces $\left[\tilde{f}_{h'}\right]_{h'}$ with $\mathsf{nullimport}\ h'$ and we know by $\tilde{f}_1$ being h–nice that $\mathsf{collapse}_{H^*_{h'',1}}\left(\left[\tilde{f}''\right]_{h''}\right)$ is consistent with the expansion of h'' in C_h. We are concerned about showing that $\mathsf{collapse}_{H^*_{h'',2}}\left(\left[\tilde{f}'''\right]_{h''}\right)$ is consistent with the expansion of h'' in C_h as well. Here $\tilde{f}''' = \tilde{f}''\left[\left[\tilde{f}_{h'}\right]_{h'} \mapsto \mathsf{nullimport}\ h'\right]$. Let H_{extr} be the set of header names whose expansions were extracted from the $\tilde{f}''$ by the application of $\restriction_h$. Suppose it were the case that the difference, $H^*_{h'',1} \setminus H^*_{h'',2}$, was such that it was entirely contained in H_{extr}. Then we would know that the collapse under $H^*_{h'',1}$ of $\tilde{f}''$ would be the same as collapsing under $H^*_{h'',2}$ of $\tilde{f}'''$, because the additional headers that were collapsed under $H^*_{h'',1}$ are all part of H_{extr} and thus their expansion is not in $\tilde{f}'''$. But H_{new}, by definition, is the set of headers whose expansion has been moved out of $\tilde{f}_{h''}$, and thus equal to H_{extr}. Also, we know that $H^*_{h'',1} \setminus H^*_{h'',2} \subseteq H_{\text{new}}$.

(b) otherwise i.e. $\left[\tilde{f}_{h'}\right]_{h'} \notin \left[\tilde{f}''\right]_{h''}$: ($\mathsf{nullimport}\ h' \in \tilde{f}''$ is possible) Such a h'' cannot contain the expansion of any $h''' \in H_{\text{new}}$ because the unique expansion of h''' is in $\tilde{f}_{h'}$. And since $H^*_{h'',1} \setminus H^*_{h'',2} \subseteq H_{\text{new}}$ it is trivially the case that if $\mathsf{collapse}_{H^*_{h'',1}}\left(\tilde{f}''\right)$ was consistent then $\mathsf{collapse}_{H^*_{h'',2}}\left(\tilde{f}''\right)$ is consistent.

Subproof for $H^*_{h'',1} \setminus H^*_{h'',2} \subseteq H_{\mathbf{new}}$ We forget the restriction for the time being and from Eq. (3.4) and (3.5) observe that $U^*_{\tilde{f}_1} \setminus U^*_{\tilde{f}_2}$ is of the form $(C_1 \setminus I_1) \setminus (C_2 \setminus I_2)$, where C_i and I_i are the terms $\mathsf{close}_{\tilde{f}_i}\left(H_{\tilde{f}_i}\right)$ and $\left\{h' \,\middle|\, \mathsf{import}\ h' \in \tilde{f}_{h,i};\ \left[\tilde{f}_{h,i}\right]_h \in \tilde{f}_i\right\}$, respectively. With de-morgans law and distributing the intersection we can see it as $((C_1 \setminus C_2) \setminus I_1) \cup ((I_2 \setminus I_1) \cap C_1))$. Therefore $U^*_{\tilde{f}_1} \setminus U^*_{\tilde{f}_2} \subseteq (C_1 \setminus C_2) \cup (I_2 \setminus I_1)$ and if we show subsumption by H_{new} and argue about the restriction then we will be done. We examine the two differences in turn.

We start by examining the term corresponding to $I_2 \setminus I_1$ and show that $I_2 \setminus I_1 \in H_{\text{new}}$. $I_2 \setminus I_1$ is the number of import statements added to $\left[\tilde{f}_h\right]_h$. Then it is trivially true that all the added imports are due to the ones in $\left[\tilde{f}_{h'}\right]_{h'}$. But this is *exactly* the set H_{new}.

Now we examine the term corresponding to $C_1 \setminus C_2$ and show that $C_1 \setminus C_2 \in H_{\text{new}}$. But the difference is *at most* the header, h', being swapped, i.e. $\mathsf{close}_{\tilde{f}_1}\left(H_{\tilde{f}_1}\right) \setminus \mathsf{close}_{\tilde{f}_2}\left(H_{\tilde{f}_2}\right) \subseteq \{h'\}$. This can easily be inferred from the observations on what an application of $\restriction_h$ means in terms of the nullorder tree in the proof of the previous lemma. Therefore again, $C_1 \setminus C_2$ is contained in H_{new}.

Lastly, observe that if we discount consideration of headers that were deleted, $\restriction_h$ is order preserving. In other words, if h_1 and h_2 appear in both $U^*_{\tilde{f}_1}$ and $U^*_{\tilde{f}_2}$ then they appear in the same order in both. This is formally proved in Observation 3.15. This is easy to see using the argument above about what an application of $\restriction_h$ means in terms of the nullorder tree. It is then trivial to observe that for any h'' that appears in both $U^*_{\tilde{f}_1}$ and $U^*_{\tilde{f}_2}$, if $U^*_{\tilde{f}_1} \setminus U^*_{\tilde{f}_2} \subseteq H_{\text{new}}$ then $U^*_{\tilde{f}_1}|_{<h''} \setminus U^*_{\tilde{f}_2}|_{<h''} \subseteq H_{\text{new}}$. □

In the proof of the following we will use the notion of a statement $\tilde{s} \in \tilde{f}$ *corresponding* to another statement $s \in \mathcal{F}(h)$ for some particular h which is expanded in $\tilde{f}$. Since each h has a unique expansion, $s \in \mathcal{F}(h)$ is either preprocessed in a unique location or is not preprocessed at all. In case it is preprocessed then the unique location where that happens, $\tilde{s}$, is the place that *corresponds* to s. Thus, a set of statements that are preprocessed are in one to one correspondence with statements in the trace.

Another notion that we will employ is that of a statement being *reachable* from a particular portion of a trace by which we mean the following: Suppose there is a trace $\tilde{f} = \tilde{f}_a, \tilde{f}_b$ then statement $s \in \tilde{f}$ is reachable from $\tilde{f}_b$ if $s \in \tilde{f}_b$ or if there exists $\mathsf{nullimport}\ h \in \tilde{f}_b$ such that if we go to $\left[\tilde{f}_h\right]_h \in \tilde{f}$ and then examine $\tilde{f}_h$ and keep doing this iteratively then we will reach s. Notice that $\left[\tilde{f}_h\right]_h$ necessarily occurs before $\mathsf{nullimport}\ h$ in the trace and therefore we always traverse towards the front of the trace.

Lemma 3.17 *If $\Delta_g; \mathcal{F} \vdash \mathcal{R}_3(f)$ and there are fragment reductions as follows:*

$$\mathcal{F} \vdash \langle \cdot; \cdot; \Delta_g; f\rangle \longrightarrow^* \left\langle \tilde{f}_x \left[\tilde{f}_h\right]_h \tilde{f}_y; \mathcal{I}_{\tilde{f}}; \Delta; \cdot \right\rangle \tag{3.6}$$

$$\mathcal{F} \vdash \langle \cdot; \cdot; \Delta_g; f\rangle \longrightarrow^* \left\langle \tilde{f}', \mathsf{import}\ h'; \mathcal{I}_{\tilde{f},h'}; \Delta'; f' \right\rangle \longrightarrow^* \left\langle \tilde{f}' \left[\tilde{f}_{h'}\right]_{h'}; \mathcal{I}'_{\tilde{f},h'}; \Delta'_2; f'_2 \right\rangle \tag{3.7}$$

$$\mathcal{F} \vdash \langle \cdot; \cdot; \Delta_g; \mathcal{F}(h), \mathsf{end}\ h\rangle \longrightarrow^* \left\langle \tilde{f}_{C_h}, \mathsf{import}\ h'; \mathcal{I}_{C_h,h'}; \Delta''; f'' \right\rangle \tag{3.8}$$

Let $\tilde{f} = \tilde{f}_x \left[\tilde{f}_h\right]_h \tilde{f}_y$ and let $H^*_{h'}$ be as defined by Eq. 3.5 in Definition 3.14b, i.e.:

$$\begin{aligned} H &= \left\langle h' \,\middle|\, \mathsf{nullimport}\ h' \in \tilde{f}_h \right\rangle \\ U^* &= \mathsf{close}_{\tilde{f}}(H) \setminus \left\{h' \,\middle|\, \mathsf{import}\ h' \in \tilde{f}_h\right\} \\ H^*_{h'} &= U^*|_{<h'} \end{aligned}$$

then $\mathcal{I}_{C_h,h'} \subseteq H^*_{h'} \cup \mathcal{I}_{\tilde{f},h'} \cup H_x$, for some H_x such that $\forall x \in H_x\ .\ \left[\tilde{f}_x\right]_x \notin \left[\tilde{f}_{h'}\right]_{h'}$.

Proof By the arguments made in previous lemma we know that under $\Delta_g; \mathcal{F} \vdash \mathcal{R}_3(f)$ and $h \in \mathcal{I}_{\tilde{f}}$ it is the case that each statement preprocessed in C_h is also preprocessed in $\tilde{f}$. Then let us start by examining the cases for the relative positioning of $\mathsf{import}\ h'$ (Eq. (3.7)) with respect to $\tilde{f}_x, \tilde{f}_h$ and $\tilde{f}_y$ (Eq. (3.6)), which we can do because they both occur in the same preprocessing: (a) $\mathsf{import}\ h' \in \tilde{f}_x$: possible but to avoid recursion ($\mathsf{import}\ h'$ is seen inside of $\tilde{f}_h$ since it is seen inside of C_h), we need it to close before h starts. (b) $\mathsf{import}\ h' \in \tilde{f}_h$: certainly possible, but means that it would have to close inside as well. (c) $\mathsf{import}\ h' \in \tilde{f}_y$: Not possible because the $\mathsf{import}\ h'$ inside of $\tilde{f}_h$ is seen and therefore this situation would violate strict well formedness. Therefore the possible structures for $\tilde{f}$ are:

(a) $\overbrace{\tilde{f}_p \left[\tilde{f}_{h'}\right]_{h'} \tilde{f}_q}^{\tilde{f}_x} \overbrace{\left[\tilde{f}_r, \mathsf{nullimport}\ h' \ldots\right]_h}^{[\tilde{f}_h]_h} \tilde{f}_y$: In this case we need to argue that if $h'' \in \mathcal{I}_{C_h,h'}$ then $h'' \in \mathcal{I}_{\tilde{f},h'} \cup H^*_{h'} \cup H_x$. $h'' \in \mathcal{I}_{C_h,h'}$ implies that $\mathsf{import}\ h'' \in \tilde{f}_{C_h}$ which means that the corresponding $\mathsf{import}\ h''$ is reached before $\mathsf{import}\ h'$ is preprocessed. The $\mathsf{import}\ h'$ statement reduces to the $\mathsf{nullimport}\ h'$ as shown and therefore $\mathsf{import}\ h''$ is reduced in the production of the trace before $\mathsf{nullimport}\ h'$, i.e. in $\left(\tilde{f}_p \tilde{f}_{h'} \tilde{f}_q \tilde{f}_r\right)$. Also, $\mathsf{import}\ h''$ has to be reachable from $\tilde{f}_r$ because $\mathsf{import}\ h'' \in \tilde{f}_{C_h}$.

Since h'' is preprocessed before $\mathsf{nullimport}\ h'$, we now worry about which particular trace it appears in. If it appears in $\tilde{f}_p$ then $h'' \in \mathcal{I}_{\tilde{f},h'}$ (and also H_x), and we are done. If it appears in $\tilde{f}_q$ or $\tilde{f}_r$ then $h'' \in H_x$ and we are done. The last case is of it appearing in $\tilde{f}_{h'}$. As seen at the end of the last paragraph, $\mathsf{import}\ h''$ is reachable from $\tilde{f}_r$ because $\mathsf{import}\ h'' \in \tilde{f}_{C_h}$. But for h'' to appear *strictly before* $\mathsf{nullimport}\ h''$ it has to be the case that a $\mathsf{nullimport}\ h''$ is reachable from $\tilde{f}_r$. Therefore, $h'' \in H^*_{h'}$. To better see the last argument, it can be easily seen that a structure of the form $\left[\ldots\left[\ldots\right]_{h''}\ldots\right]_{h'} \tilde{f}'_q \left[\tilde{f}'_r, \mathsf{nullimport}\ h' \ldots\right]_h$, where $\tilde{f}'_q$ and $\tilde{f}'_r$ do not contain a reference to h'' cannot lead to h'' being reachable from $\tilde{f}'_r$.

(b) $\tilde{f}_x \overbrace{\left[\ldots\left[\tilde{f}_{h'}\right]_{h'}\ldots\right]_h}^{[\tilde{f}_h]_h} \tilde{f}_y$: Since all $\mathsf{import}\ h'$ statements seen in C_h are also seen in $\tilde{f}_h$. Therefore if $\left[\tilde{f}_{h'}\right]_{h'} \in \tilde{f}_h$ then it has to be the case that it is the expansion of the first $\mathsf{import}\ h'$, which was expactly the one that expanded in C_h. So the expansion of h' is at exactly the location as it was in C_h and therefore it will follow that $\mathcal{I}_{C_h,h'} \subseteq \mathcal{I}_{\tilde{f},h'}$. We have inclusion instead of equality because the set of includes seen might not be empty at the end of preprocessing for $\tilde{f}_x$.

□

Lemma 3.18 *If* $\mathcal{F} \vdash \langle \cdot; \cdot; \Delta_g; f\rangle \longrightarrow^* \left\langle \tilde{f}; \mathcal{I}; \Delta; \cdot \right\rangle$ *and* $\Delta_g; \mathcal{F} \vdash \mathcal{R}_3(f)$ *then* $\forall h \in \mathcal{I}\ .\ \tilde{f}$ *is* h–nice.

Proof There are two conditions for $\tilde{f}$ to be h–nice. We prove each in turn:

3.14a Let $f_1 = f$ and $f_2 = \mathcal{F}(h), \mathsf{end}\ h$. Refering back to Lemma 3.10, $\mathcal{I}_a = \emptyset$, $\tilde{f}_a = \cdot$ and by $h \in \mathcal{I}$ we have that $\mathcal{I}_x$ is well defined and hence vacuously, $\mathcal{I}_a \subseteq \mathcal{I}_x$. Also, $\Delta_g; \mathcal{F} \vdash \mathcal{R}_3(f_1)$ holds by assumption and the last assumption is trivially satisfied by the fact that $\tilde{f}_a = \cdot$. Therefore the consequence, $\tilde{f}_h$ is consistent with C_h, in Eq. (3.10), holds.

3.14b For a given $h \in \mathcal{I}$, we consider an arbitrary $h' \in U^*$ as defined in Eq. (3.5). Let us use $\mathcal{I}_{C_h}$ and $\mathcal{I}_{\tilde{f}}$ to indicate the includes that were seen upto the point of encountering the first $\mathsf{import}\ h'$ in the respective preprocessings. And let $\left[\tilde{f}_1\right]_{h'} \in \tilde{f}$ and $\left[\tilde{f}_2\right]_{h'} \in C_h$ be the expansions of h' is the traces as indicated.

Formally,

$$\begin{aligned}\mathcal{F} \vdash \langle \cdot;\cdot;\Delta_g; f\rangle &\longrightarrow^* \left\langle \tilde{f}_a, \mathsf{import}\ h'; \mathcal{I}_{\tilde{f}}; \Delta_a; f_a \right\rangle \longrightarrow^* \left\langle \tilde{f}_a, \left[\tilde{f}_1\right]_{h'}; \mathcal{I}_b; \Delta_b; f_b \right\rangle \\ \mathcal{F} \vdash \langle \cdot;\cdot;\Delta_g; \mathcal{F}(h), \mathsf{end}\ h\rangle &\longrightarrow^* \left\langle \tilde{f}_x, \mathsf{import}\ h'; \mathcal{I}_{C_h}; \Delta_x; f_x \right\rangle \longrightarrow^* \left\langle \tilde{f}_x, \left[\tilde{f}_2\right]_{h'}; \mathcal{I}_y; \Delta_y; f_y \right\rangle\end{aligned}$$

We cannot directly ensure that $\mathcal{I}_{C_h} \subseteq \mathcal{I}_{\tilde{f}}$ (which is the $\mathcal{I}_a \subseteq \mathcal{I}_x$ condition that we need for Eq.(3.10)). But if we let $H^*_{h'}$ be the include names ahead of h' and let H_x be an arbitrary set of headers names whose expansion does not appear in $\tilde{f}_1$, both as defined in Eq. (3.5) and exactly as in Lemma 3.17, then the lemma states that $\mathcal{I}_{C_h} \subseteq H^*_{h'} \cup \mathcal{I}_{\tilde{f}} \cup H_x$. Also, $\Delta_g; \mathcal{F} \vdash \mathcal{R}_3(f_1)$ holds by assumption. By the fact that $h' \in U^*$, we know that the expansion of h' appears before the expansion of h in $\tilde{f}$ and $\Delta_g; \mathcal{F} \vdash \mathcal{R}_3\left(\tilde{f}\right)$ ensure that all changes in the expansion of h are disjoint from the uses in the expansion of h'. And therefore the last assumption holds. Therefore, by Corollary 3.13, $\mathsf{collapse}_{H^*_{h'} \cup H_x}\left(\tilde{f}_1\right)$ is consistent with $\tilde{f}_2$. And by the definition of H_x and Corollary 3.12 we get that $\mathsf{collapse}_{H^*_{h'}}\left(\tilde{f}_1\right)$ is consistent with $\tilde{f}_2$. □

Definition 3.19 (Distance of $\tilde{f}$ from h) *Let U^* be as defined by Eq. (3.4) in Definition 3.14. Then the* import distance, *$\delta_h\left(\tilde{f}\right)$ of a trace $\tilde{f}$ from a header h is*

$$\delta_h\left(\tilde{f}\right) = \begin{cases} |U^*| & \mathsf{import}\ h \in \tilde{f}, \\ 0 & \textit{otherwise} \end{cases} \tag{3.9}$$

Lemma 3.20 *If $\tilde{f}_1 \upharpoonright_h \tilde{f}_2$ and $\delta_h\left(\tilde{f}_1\right) = d$ then $\delta_h\left(\tilde{f}_2\right) \leq d - 1$.*

Proof Trivial by observation on the definitions of the distance function (Eq. (3.9)) and that of U^* (Eq. (3.4)) and the operation performed by $\upharpoonright_h$. □

Lemma 3.21 *If $\delta_h\left(\tilde{f}\right) \neq 0$ then $\upharpoonright_h$ can take a step.*

Proof Since $\delta_h\left(\tilde{f}\right) \neq 0$ implies by definition that there is at least one element in U^*. Suppose h' be one of those elements. From the definition of U^* (Eq. (3.4)) we know that this implies that $\left[\tilde{f}_{h'}\right]_{h'} \notin \left[\tilde{f}_h\right]_h$, while there is some h'' for which $\mathsf{nullimport}\ h'' \in \tilde{f}_h$ and which makes $\mathsf{nullimport}\ h'$ reachable (because that is the only way h' can be in $\mathsf{close}_{\tilde{f}}(H)$). If $\left[\tilde{f}_{h''}\right]_{h''} \notin U^*$ then by the fact that $\mathsf{nullimport}\ h'' \in \tilde{f}_h$ we know that $\left[\tilde{f}_{h''}\right]_{h''} \in \tilde{f}_h$. This implies that the $\mathsf{nullimport}$ statement for all its children is in $\tilde{f}_h$. We can keep repeating this argument (and coming down the nullorder tree) until we hit h' or an ancestor, h_a, of h' for which the $\mathsf{nullimport}\ h_a$ is in $\tilde{f}_h$ and $\left[\tilde{f}_{h_a}\right]_{h_a}$ is outside. It has to be the case that the expansion for h_a is ahead of $\tilde{f}_h$. Therefore we know that there is at least one, if not more, header names which satisfies the structure requirements in [Transfer Function]. If there are more then there has to be a first for which the rule can be applied. □

Lemma 3.22 *If $\mathsf{import}\ h \in \tilde{f}$ and $\delta_h\left(\tilde{f}\right) = 0$ and $\tilde{f}$ is h–nice then $\tilde{f} = \tilde{f}_a\,[C_h]_h\,\tilde{f}_b$.*

Proof From $\tilde{f}$ is h–nice we know that $\tilde{f} = \tilde{f}_a\left[\tilde{f}_h\right]_h \tilde{f}_b$ where $\tilde{f}_h$ is consistent with C_h. Then from the condition for consistency, we know that either $\tilde{f}_h = C_h$ or $\exists h'$ such that $\tilde{f}_h = \tilde{f}_{1a}, \mathsf{nullimport}\ h', \tilde{f}_{1b}$ and $C_h = \tilde{f}_{2a}, \mathsf{import}\ h', \tilde{f}'_h, \mathsf{end}\ h', \tilde{f}_{2b}$, where $\tilde{f}_{1a}, \tilde{f}_{1b}$ are consistent with $\tilde{f}_{2a}, \tilde{f}_{2b}$ respectively. We show by contradiction that the latter cannot happen.

Since $\mathsf{import}\ h \in \tilde{f}$, the first case in Eq. (3.9) applies. From $\delta_h\left(\tilde{f}\right) = 0$ and $\mathsf{nullimport}\ h' \in \tilde{f}_h$ we infer, from Eq. (3.3) and (3.4), that $h' \in H$ which implies $h' \in \mathsf{close}_{\tilde{f}}(H)$ and therefore $\mathsf{import}\ h' \in \tilde{f}_h$. $\mathsf{import}\ h' \in \tilde{f}_h$ means that one of $\tilde{f}_{1a}$ or $\tilde{f}_{1b}$ contains $\mathsf{import}\ h'$. Because of well formedness we know that there is a unique expansion of h' and therefore a single occurrence of $\mathsf{import}\ h'$ in C_h. Therefore $\tilde{f}_{2a}$ and $\tilde{f}_{2b}$, cannot contain an $\mathsf{import}\ h$; which implies that consistency of the enclosing subtraces is not possible and hence the contradiction.

□

Lemma 3.23 (Consistent Interpretation) *If*

- *the fragment reduces:* $\mathcal{F} \vdash \langle \cdot; \cdot; \Delta_g; f\rangle \longrightarrow^* \left\langle \tilde{f}; \mathcal{I}; \Delta; \cdot \right\rangle$ *and* $\left\langle \mathcal{A}_\emptyset; \tilde{f} \right\rangle \longrightarrow^* \langle \mathcal{A}; \cdot\rangle$
- *there is order independence:* $\Delta_g; \mathcal{F} \vdash \mathcal{R}_3(f)$
- *there exists an import of* h *in the reduction:* $\mathsf{import}\ h \in \tilde{f}$
- *If* $f_h = \mathcal{F}(h), \mathsf{end}\ h$ *and* $\mathcal{F} \vdash \langle \cdot; \cdot; \Delta_g; f_h\rangle \longrightarrow^* \left\langle \tilde{f}_h; \mathcal{I}_h; \Delta_h; \cdot \right\rangle$ *and* $\left\langle \mathcal{A}_\emptyset; \tilde{f}_h \right\rangle \longrightarrow^* \langle \mathcal{A}_h; \cdot\rangle$

then $\mathcal{A}_h \subseteq \mathcal{A}$.

Proof From Lemma 3.18 we know that if $\Delta_g; \mathcal{F} \vdash \mathcal{R}_3(f)$ then $\forall h\ .\ \mathsf{import}\ h \in \tilde{f}\ .\ \tilde{f}$ is h–nice. Then by Lemma 3.16 we know that the traces obtained by repeated applications of $\restriction_h$ remain h–nice. Suppose $\delta_h\left(\tilde{f}\right) = d$ then by Lemma 3.20 we know that at most d applications suffice to obtain $\tilde{f}'$ such that $\delta_h\left(\tilde{f}'\right) = 0$. By Lemma 3.22 we know that $\tilde{f}' = \tilde{f}_a\,[C_h]_h\,\tilde{f}_b$. Let $\left\langle \mathcal{A}_\emptyset; \tilde{f}' \right\rangle \longrightarrow^* \left\langle \mathcal{A}_{\tilde{f}'}; \cdot \right\rangle$, then from Corollary 3.5 we know that $\mathcal{A} = \mathcal{A}_{\tilde{f}'}$. Also from Lemma 3.6 we know that $\mathcal{A}_h \subseteq \mathcal{A}_{\tilde{f}'} (= \mathcal{A})$ □

4 Consistent Typing and Type Safe Linking

Given the lemma of consistent interpreation from the previous section, we can now show that if two fragments satisfy the conditions of [Rule 1] and the commonly-included header is order-independent, then any symbol exported by one and imported by the other has the same type in both.

Lemma 4.1 (Consistent Typing) *Let fragment* f_e *export* g *and let fragment* f_i *import* g*, and let both* f_e *and* f_i *include a common header fragment* $f_h (= \mathcal{F}[h])$ *that declares the variable:*

$$\begin{array}{lll} \Delta; \mathcal{F} \vdash f_e \rightsquigarrow \mathcal{A}_e; \mathcal{I}_e, & g \in \mathcal{A}_e^E, & h \in \mathcal{I}_e \\ \Delta; \mathcal{F} \vdash f_i \rightsquigarrow \mathcal{A}_i; \mathcal{I}_i, & g \in \mathcal{A}_i^I, & h \in \mathcal{I}_i \\ \Delta; \mathcal{F} \vdash f_h \rightsquigarrow \mathcal{A}_h; \mathcal{I}_h, & g \in \mathcal{A}_h^D & \end{array}$$

Then if

$$\begin{array}{ll} \Delta; \mathcal{F} \vdash f_e \xrightarrow{comp} O_e & \Delta; \mathcal{F} \vdash \mathcal{R}_3(f_e) \\ \Delta; \mathcal{F} \vdash f_i \xrightarrow{comp} O_i & \Delta; \mathcal{F} \vdash \mathcal{R}_3(f_i) \end{array}$$

all hold, then $\mathcal{A}_h^N(g) = \mathcal{A}_e^N(g) = \mathcal{A}_i^N(g)$*, i.e., g maps to the same type in each fragment.*

Proof By assumption, preprocessing f_e and f_i will eventually preprocess the statement $s = \mathsf{include}\ h$. Thus we have:

$$\begin{array}{l} \mathcal{F} \vdash \langle \circ; \mathcal{A}_\emptyset; \Delta; f_e\rangle \longrightarrow^* \langle h_{1e}; \mathcal{A}_{1e}; \Delta_{1e}; (s, f_{2e})\rangle \\ \mathcal{F} \vdash \langle h_{1e}; \mathcal{A}_{1e}; \Delta_{1e}; (s, f_2)\rangle \longrightarrow^* \langle h_{1e}; \mathcal{A}_{2e}; \Delta_{2e}; f_{2e}\rangle \\ \mathcal{A}_{2e} \subseteq \mathcal{A}_e \end{array}$$

Then by Lemma 3.23, we have $\mathcal{A}_h^N \subseteq \mathcal{A}_{2e}^N$. But then since $\mathcal{A}_{2e}^N \subseteq \mathcal{A}_e^N$, we have $\left(g \mapsto \mathcal{A}_h^N(g)\right) \in \mathcal{A}_e^N$. The by [Compile], we have $\vdash \mathcal{A}_e^N$, and therefore $\mathcal{A}_e^N(g) = \mathcal{A}_h^N(g)$. Similarly we can show that $\mathcal{A}_i^N(g) = \mathcal{A}_h^N(g)$, because f_i included the same file. □

Theorem 4.2 (Type-Safe Linking) *Suppose* $\Delta; \mathcal{F} \vdash \mathcal{R}(\mathcal{P})$*, and suppose* $\Delta; \mathcal{F} \vdash \mathcal{P} \xrightarrow{comp} [\emptyset \Rightarrow H_{\mathcal{P}} : \Psi_{E\mathcal{P}}]$*. Also suppose that for any* $f_i, f_j \in \mathcal{P}$ *that are distinct* $(i \neq j)$*, it is the case that*

$$\Delta; \mathcal{F} \vdash f_i \xrightarrow{comp} [\Psi_{Ii} \Rightarrow H_i : \Psi_{Ei}]$$
$$\Delta; \mathcal{F} \vdash f_j \xrightarrow{comp} [\Psi_{Ij} \Rightarrow H_j : \Psi_{Ej}]$$
$$\Delta; \mathcal{F} \vdash [\Psi_{Ii} \Rightarrow H_i : \Psi_{Ei}] \circ [\Psi_{Ij} \Rightarrow H_j : \Psi_{Ej}] \xrightarrow{comp} O_{ij}$$

Then

$$\vdash [\Psi_{Ii} \Rightarrow H_i : \Psi_{Ei}] \text{ link } [\Psi_{Ij} \Rightarrow H_j : \Psi_{Ej}] \rightsquigarrow O_{ij}$$

Proof By assumption [Link] holds. Observe that the linked file form (O_{ij}) in [Link] is the same as in [MTAL$_0$-Obj], so we just need to show the hypotheses of this rule. To show [MTAL$_0$-Obj], we first need to show each object file is well-formed, which follows by Lemma 2.7. The last premise of [MTAL$_0$-Obj], disjointness of the domains of H_i and H_j, is the same as the premise of [Link], so that also holds. Thus we only need to show link compatibility, or $\vdash [\Psi_{Ii} \Rightarrow H_i : \Psi_{Ei}] \overset{\text{lc}}{\leftrightarrow} [\Psi_{Ij} \Rightarrow H_j : \Psi_{Ej}]$. We show each premise of [MTAL$_0$-LC] in turn.

- $\text{dom}(\Psi_{Ei}) \cap \text{dom}(\Psi_{Ej}) = \emptyset$. Notice $\text{dom}(\Psi_{Ei}) = E_i$ by [Compile] and $E_i \subseteq \text{dom}(H_i)$, which we observe holds because the rules in Figure 7 and Figure 9 only add symbols to E that are also added to $\text{dom}(H)$ (see rule [Let]). Similarly, $\text{dom}(\Psi_{Ej}) = E_j \subseteq \text{dom}(H_j)$. Then since by [Link] we have $\text{dom}(H_i) \cap \text{dom}(H_j) = \emptyset$, we have $\text{dom}(\Psi_{Ei}) \cap \text{dom}(\Psi_{Ej}) = \emptyset$.

- $\vdash \Psi_{Ii} \sim \Psi_{Ej}$. By assumption, we have $\Delta; \mathcal{F} \vdash \mathcal{R}(\mathcal{P})$. Thus by [All], we have in particular

$$\Delta; \mathcal{F} \vdash \mathcal{R}_1(f_i, f_j)$$
$$\Delta; \mathcal{F} \vdash \mathcal{R}_3(f_i) \quad \Delta; \mathcal{F} \vdash \mathcal{R}_3(f_j)$$

  Let $\mathcal{A}_i$ and $\mathcal{A}_j$ are the accumulators from the preprocessing of f_i and f_j, respectively. Now consider some g in $\text{dom}(\Psi_{Ii}) \cap \text{dom}(\Psi_{Ej})$. Then we have $g \in \mathcal{A}_i^I$ and $g \in \mathcal{A}_j^E$. Further, by [Rule 1] we have $\Delta; \mathcal{F} \vdash g \xleftarrow{\text{decl}} \mathcal{I}_i \cap \mathcal{I}_j$. Then by [Sym-Decl] there exists some $h \in \mathcal{I}_i \cap \mathcal{I}_j$ such that $\Delta; \mathcal{F} \vdash \mathcal{F}(h) \rightsquigarrow \mathcal{A}; \mathcal{I}_h$ and $g \in \mathcal{A}^D$. Then we can apply Lemma 4.1 to yield $\mathcal{A}_i^N(g) = \mathcal{A}_j^N(g)$. But then we have $\Psi_{Ii}(g) = \Psi_{Ej}(g)$ by [Compile], and therefore $\vdash \Psi_{Ii} \sim \Psi_{Ej}$ holds.

- $\vdash \Psi_{Ij} \sim \Psi_{Ei}$. Symmetric argument to the previous case.

- $\vdash \Psi_{Ii} \sim \Psi_{Ij}$. Let us consider some $g \in \text{dom}(\Psi_{Ii}) \cap \text{dom}(\Psi_{Ij})$. Then we assume that $\exists f_k \in \mathcal{P}$ s.t. $\Delta; \mathcal{F} \vdash f_k \xrightarrow{\text{comp}} [\Psi_{Ik} \Rightarrow H_k : \Psi_{Ek}]$ and $g \in E_k$, i.e., we assume that some fragment exports the symbol g, because we assumed the fully-compiled program had no unresolved symbols. Then by the same argument as the previous two cases we can conclude $\vdash \Psi_{Ii} \sim \Psi_{Ek}$ and $\vdash \Psi_{Ij} \sim \Psi_{Ek}$, since the Cmod rules hold for the whole program. Then we have $\Psi_{Ek}(g) = \Psi_{Ii}(g)$ and $\Psi_{Ek}(g) = \Psi_{Ij}(g)$, and therefore $\Psi_{Ii}(g) = \Psi_{Ij}(g)$.

□

5 Information Hiding

First, we can prove that any symbol not in a header file is never imported, and thus is private.

Theorem 5.1 (Global Variable Hiding) *Suppose* $\Delta; \mathcal{F} \vdash \mathcal{R}(\mathcal{P})$*, suppose* $\Delta; \mathcal{F} \vdash \mathcal{P} \xrightarrow{comp} [\emptyset \Rightarrow H_{\mathcal{P}} : \Psi_{E\mathcal{P}}]$*, and suppose for all* $f_i \in \mathcal{P}$ *we have* $\Delta; \mathcal{F} \vdash f_i \rightsquigarrow \mathcal{A}_{fi}; \mathcal{I}_i$*, and for all* $h_j \in \bigcup_i \mathcal{I}_i$ *that* $\Delta; \mathcal{F} \vdash \mathcal{F}(h_j) \rightsquigarrow \mathcal{A}_{hj}$*. Then for all* $f_i \in \mathcal{P}$*,* $g \notin \bigcup_j \mathcal{A}_{hj}^D$ *implies* $g \notin \Psi_{Ii}$ *where* $\Delta; \mathcal{F} \vdash f_i \xrightarrow{comp} [\Psi_{Ii} \Rightarrow H_i : \Psi_{Ei}]$*.*

This theorem says that if $\mathcal{P}$ obeys the Cmod rules and includes headers h_j, then any symbol g that is not in $\mathcal{A}_{hj}^D$ for any j (i.e., is not declared in any header file) is never imported.

For type names, we can prove a related property: any type name owned by a source fragment (a code file) has no concrete type in any other fragment.

Theorem 5.2 (Type Definition Hiding) *Suppose* $\Delta; \mathcal{F} \vdash \mathcal{R}(\mathcal{P})$*, and for some* $f_i \in \mathcal{P}$ *we have* $\Delta; \mathcal{F} \vdash f_i \rightsquigarrow \mathcal{A}_i; \mathcal{I}_i$*. Further suppose that* $(t \mapsto \tau^\circ) \in \mathcal{A}_i^T$*. Then for any fragment* $f_j \in \mathcal{P}$ *such that* $f_i \neq f_j$ *and* $\Delta; \mathcal{F} \vdash f_j \rightsquigarrow \mathcal{A}_j$*, we have* $t \notin \mathrm{dom}\left(\mathcal{A}_j^T\right)$*. Additionally, if* $(t \mapsto \tau^h) \in \mathcal{A}_i^T$*, then* $h \in \mathcal{I}_i$*.*

This theorem says that if $\mathcal{P}$ obeys the CMOD rules and contains fragment f_i, then any type t owned by f_i is not owned by any other fragments $f_j \neq f_i$. Additionally, if a type is defined in a header then a fragment only sees the definition if it includes the header. Together, Theorems 5.1 and 5.2 give us information hiding.

References

[1] N. Glew and G. Morrisett. Type-safe linking and modular assembly language. In *POPL*, 1999.

[2] S. Srivastava, M. Hicks, J. S. Foster, and P. Jenkins. Modular information hiding and type safety for C, June 2007. Submitted to IEEE Transactions on Software Engineering. Full version of TLDI 07 paper. Available at `http://www.cs.umd.edu/~mwh/papers/cmod-journal.pdf`.