Formal definition and proof of completeness of inlining

Abstract

We prove a dual result to the verification-preserving inlining theorem: If the inlined program verifies, then there is no true error for bounded executions of the original program. We call this result *completeness of inlining* (or simply *completeness*). In our appendix of the supplementary material (see appendix.pdf), we have shown the key ideas behind the completeness result and its proof. This document shows the details of the proof. We reuse the definitions and lemmas from the Isabelle/HOL formalization of the verification-preserving inlining proof. Section 2 formalizes the restrictions described in our appendix: The bounded semantics and the well-annotated hypothesis. Section 3 formally expresses the theorem, and the stronger property that we prove by induction. Finally, each section afterwards proves a case of the proof by induction.

Contents

1 Differences in notations with the paper

This document uses the notations from Isabelle to express the semantics, which are slightly different than the ones from the paper, which were used for the sake of presentation. In this document, annotations are part of a program: While loops are now expressed as **while** (b) **inv** I $\{s\}$, where I is the loop invariant. Method pre- and postconditions are parts of the set of methods M. Note that inlining ignores annotations, since the inlined program contains no method call and no loop.

Moreover, the function *sem* takes as input a set of states and a statement, and returns a set of states

$$sem_M : \mathcal{P}(\Sigma) \times Stmt \to \mathcal{P}(\Sigma)$$

Furthermore, $sem_M(A, s)$ is undefined if there exists a state in A for which s does not verify.

Similarly, the function *ver* takes as input a set of states and a statement, and returns a boolean:

$$ver_M : \mathcal{P}(\Sigma) \times Stmt \to Boolean$$

$ver_M(A, s)$ holds if and only if the statement s verifies in all states of A.

2 Restrictions

2.1 Bounded semantics

As explained in the appendix, we need *bounded semantics*, that is semantics which stop executions when the parameter n of the inlining reaches 0. We showed an example with instrumentation in the appendix. We define here bounded semantics which behave in a similar way as this instrumentation.

The states φ we consider for the execution all define a special variable *dep*, which we also represent as a property of a state given by the following function:

$$dep : \Sigma \mapsto \mathbb{N}$$

This special variable *dep* cannot be modified by most statements. It is only modified by the bounded semantics itself, when dealing with loops and method calls.

The bounded semantics are defined with the two functions $semB_M : \mathcal{P}(\Sigma) \times Stmt \to \mathbb{P}(\Sigma)$ and $verB_M : \mathcal{P}(\Sigma) \times Stmt \to Bool$. For a set of states A and a statement s, $verB_M(A, s)$ holds if and only if s verifies with all states in A in the bounded semantics. In this case, $semB_M(A, s)$ is the set of states resulting from the execution of s from the states in A in the bounded semantics. The definitions of *verB* and *semB* are similar the ones of *ver* and *sem* for most cases, except for method calls and loops.

We use the notation **assign** b (where b is an assertion), it is syntatic sugar for **havoc** *dep* ; **assume** b. The bounded semantics of method calls is:

$$verB_M(\{\varphi\}, \vec{y} := m(\vec{x})) = \begin{cases} \top & \text{if } dep(\varphi) = 0 \\ ver_M(\{\varphi\}, dep \mathrel{-}= 1 \ ; \ \vec{y} := m(\vec{x}) \ ; \ dep \mathrel{+}= 1) & \text{otherwise} \end{cases}$$

$$semB_M(\{\varphi\}, \vec{y} := m(\vec{x})) = \begin{cases} \varnothing & \text{if } dep(\varphi) = 0 \\ sem_M(\{\varphi\}, dep \mathrel{-}= 1 \ ; \ \vec{y} := m(\vec{x}) \ ; \ dep \mathrel{+}= 1) & \text{otherwise} \end{cases}$$

and loops (where $\vec{l} := modif(s) \cap \sigma(\varphi)$),

$$\begin{aligned} &verB_M(\{\varphi\}, \textbf{while } (b) \textbf{ inv } I \ \{s\}) \\ =&verB_M(\{|\varphi|\}, \textbf{havoc } \vec{l} \ ; \textbf{ assign } 1 \leq dep \leq dep(\varphi) \ ; \textbf{ inhale } I \ ; \textbf{ assume } b \ ; dep \mathrel{-}= 1 \ ; s \ ; \textbf{ exhale } I) \\ \wedge&ver_M(\{\varphi\}, \textbf{exhale } I \ ; \textbf{ havoc } \vec{l} \ ; \textbf{ assign } 0 \leq dep \leq dep(\varphi) \ ; \textbf{ inhale } I \ ; \textbf{ assume } \neg b \ ; dep \leftarrow dep(\varphi)) \end{aligned}$$

and

$$\begin{aligned} &semB_M(\{\varphi\}, \textbf{while } (b) \textbf{ inv } I \ \{s\}) \\ =&sem_M(\{\varphi\}, \textbf{exhale } I \ ; \textbf{ havoc } \vec{l} \ ; \textbf{ assign } 0 \leq dep \leq dep(\varphi) \ ; \textbf{ inhale } I \ ; \textbf{ assume } \neg b \ ; dep \leftarrow dep(\varphi)) \end{aligned}$$

We also define the following function

$$deps : \mathbb{N} \mapsto \text{singleton of } \Sigma$$

where $deps(n)$ is the singleton containing the unique state φ that has no impure resource and defines only the special variable *dep* with the value n. Thus, $dep(\varphi) = n$ and $\sigma(\varphi) = \{deps\}$.

2.2 Well-annotated hypothesis

This hypothesis assumes that the program has already been annotated. The idea (explained in the appendix) is that, if the assertion language is expressive enough, then we can instrument any program and create such an annotation. The well-annotated hypothesis expresses that the program we have at hand (the initial statement and the set of methods) have already been annotated in this way. The formal result we prove is therefore not that there exists an annotation such that the original program verifies, but that a program annotated in the way we describe here (well-annotated), then it verifies. We do not formally prove that we can instrument a program to obtain this property, but we illustrate here how to construct such an annotation, and why it is possible.

2.2.1 Idea: Methods calls are indexed

We express the fact that it is possible to distinguish different states using some local variables with the following definition:

Definition 1. *A set of states Ind is mutually disjoint if and only if*

$$\forall i, j \in Ind.i \neq j \Longrightarrow \neg(i \# j)$$

We then define the concept of a partial annotation for a method:

Definition 2. *A partial annotation for a method m is a list of quadruples*

$$[[m, i_1, P_1, Q_1], \ldots, [m, i_k, P_k, Q_k]]$$

with $i_j \in \Sigma$ (indices, pure states), and P_j and Q_j annotations. They represent the following:

```
method m(...) returns (...)
requires i1 || ... || ik
requires i1 ==> P1
...
requires ik ==> Pk
ensures i1 || ... || ik
ensures i1 ==> Q1
...
ensures ik ==> Qk
```

A method with such an annotation verifies if and only if it verifies for all quadruples:

Lemma 1. *If the indices of the partial annotation are mutually disjoint, then the method m partially annotated verifies if and only*

$$\forall j.1 \leq j \leq k \Longrightarrow \{i_j \wedge P_j\} m \{i_j \wedge Q_j\}$$

We also have the property that, if the indices of the assertion are mutually disjoint, then inhaling or exhaling such an assertion in a state which is compatible with one index is the same as inhaling or exhaling the assertion corresponding to the relevant quadruple:

Lemma 2. *If the indices of the partial annotation P, Q are mutually disjoint, and $\varphi \# i_j$, then*

$$\begin{aligned} ver_M(\{\varphi\}, \textbf{inhale } Q) &= ver_M(\{\varphi\}, \textbf{inhale } Q_j) \\ sem_M(\{\varphi\}, \textbf{inhale } Q) &= sem_M(\{\varphi\}, \textbf{inhale } Q_j) \\ ver_M(\{\varphi\}, \textbf{exhale } P) &= ver_M(\{\varphi\}, \textbf{exhale } P_j) \\ sem_M(\{\varphi\}, \textbf{exhale } P) &= sem_M(\{\varphi\}, \textbf{exhale } P_j) \end{aligned}$$

2.2.2 Projection

We now formally define the concept of projecting a state onto a set of local variables, which is needed to express the well-annotated hypothesis.

Definition 3. *φ is a state, V a set of variables. $\pi_{\vec{V}}(\varphi)$ is the state with the same permissions, but where we removed all variables outside of $\vec{V}$.*

$$\pi_{\vec{V}}(A) = \{\pi_{\vec{V}}(\varphi) | \varphi \in A\}$$

We then express two simple lemmas, which can be proved from the Isabelle formalization axioms. If a statement does not read any of the variables in $\vec{V}$, then these variables do not influence the execution of this statement:

Lemma 3. *If $read(s) \cap \vec{V} = \varnothing$, then, for all $\varphi \in \Sigma$:*

$$\begin{aligned} ver_M(\{\varphi\}, s) &\Longleftrightarrow ver_M(\{\overline{h}_{\vec{V}}(\varphi)\}) \\ sem_M(\{\varphi\}, s) &= sem_M(\{\overline{h}_{\vec{V}}(\varphi)\}, s) \oplus \{\pi_{\vec{V}}(|\varphi|)\} \end{aligned}$$

Lemma 4. *If $read(s) \subseteq \vec{V}$, then, for all $\varphi \in \Sigma$:*

$$\begin{aligned} ver_M(\{\varphi\}, s) &\Longleftrightarrow ver_M(\{\pi_{\vec{V}}(\varphi)\}) \\ \pi_{\vec{V}}(sem_M(\{\varphi\}, s)) &= sem_M(\{\pi_{\vec{V}}(\varphi)\}, s) \end{aligned}$$

2.2.3 Collecting method annotations

To prove that the original annotated program verifies, we need to prove that all methods verify w.r.t. their annotations. We prove the completeness theorem by computational induction on the structure of the inline function. In particular, the induction hypothesis states that the methods verify w.r.t. the partial annotations which have not been tackled yet. To capture these partial annotations which have not been tackled yet, we define the *collectAn* function. This function "collects" all partial annotations which come from the remaining method calls in the program.

The variables of this function are as follows:

- M is the set of methods of the program.
- n is the depth up to which we inline.
- $\vec{U}$ is the set of variable names already used in the inlining. It is used for avoiding the capture of variables when inlining a method call and renaming it body.
- $\vec{V}$ is the set of variable names readable in the current method (since a method body does not have access to variables not passed via the arguments).
- A is the current set of states in the execution of the inlined program.
- s is the statement we inline.

Definition 4. ***collectAn***

- *Method call (depth $= n+1$)*
  $collectAn_M^{n+1}(\vec{U}, \vec{V}, A, \vec{y} := m(\vec{x})) := \{(m, ind, P_{ind}, Q_{ind})\} \cup collectAn_M^n(\vec{U} \cup modif(s'), read(s'), A, s')$
  where
  - m *corresponds to* $(\overrightarrow{args}, \overrightarrow{rets}, \_, \_, s)$ *in the set of methods* M
  - $t := (\overrightarrow{args} \cup \overrightarrow{rets}, \vec{x} \cup \vec{y}, \vec{U}, modif(s))$
  - $s' := rename_t(s)$
  - $A' := sem_M(A, inl_M^n(\vec{U} \cup modif(s'), s'))$
  - $Inh(P_{ind}) = \pi_{\overrightarrow{args} \cup \overrightarrow{rets}}(rename_{t^{-1}}(A)) \oplus deps(n)$
  - $Inh(Q_{ind}) = \pi_{\overrightarrow{args} \cup \overrightarrow{rets}}(rename_{t^{-1}}(A')) \oplus deps(n)$
- *Loop (depth $= n+1$)*
  $collectAn_M^{n+1}(\vec{U}, \vec{V}, A, \textbf{while } (b) \textbf{ inv } I \ \{s\})$
  $:= collectAn_M^n(\vec{U}, \vec{V}, f_b(A), s) \cup collectAn_M^n(\vec{U} \cup modif(s'), \vec{V}, sem_M(f_b(A), s'), \textbf{while } (b) \textbf{ inv } I \ \{s\})$
  where $s' := inl_M^n(\vec{U}, s)$ *and* $A' := sem_M(f_b(A), s')$
- *Sequential composition*
  $collectAn_M^n(\vec{U}, \vec{V}, A, s_1 \ ; \ s_2) := collectAn_M^n(\vec{U}, \vec{V}, A, s_1) \cup collectAn_M^n(\vec{U} \cup read(s_1'), \vec{V}, sem_M(A, s_1'), s_2)$
  where $s_1' := inl_M^n(\vec{U}, s_1)$
- *Non-deterministic branching*
  $collectAn_M^n(\vec{U}, \vec{V}, A, \textbf{if } (*) \ \{s_1\} \textbf{ else } \{s_2\}) := collectAn_M^n(\vec{U}, \vec{V}, A, s_1) \cup collectAn_M^n(\vec{U}, \vec{V}, A, s_2)$
- *Other cases*
  $collectAn_M^n(\vec{U}, \vec{V}, A, s) := \varnothing$ *otherwise*

The most interesting case, the method call case, states that the partial annotation corresponding to this method call is the union of the quadruple $(m, ind, P_{ind}, Q_{ind})$ with the partial annotation corresponding to the inlining of the body of this method. P_{ind} is the assertion which captures the set of states from the execution before the method call, but renamed and for and projected onto the arguments of the method m. Similarly, Q_{ind} is the assertion which captures the set of states from the execution after the method call, but renamed and for and projected onto the arguments of the method m.

2.2.4 Well-annotated: Formal definition

We finally define the well-annotated function. This function expresses that loop invariants capture the sets of states (up to the variables $\vec{U}$) from the execution before and after each loop iteration with the right indexing, and method preconditions and postconditions capture the sets of states (up to the variables $\vec{U}$) from the execution before and after the method call, also with the right indexing.

Definition 5. ***wellAnnot***

- *Method call (depth $= n+1$)*

  $$\begin{aligned} &wellAnnot_M^{n+1}(\vec{U}, \vec{V}, A, \vec{y} := m(\vec{x})) \\ \iff &(\exists (m, ind, P_{ind}, Q_{ind}) \in collectAn_M^{n+1}(\vec{U}, \vec{V}, A, \vec{y} := m(\vec{x})).A\#|ind| \wedge A'\#|ind| \wedge \sigma(|ind|) \subseteq \vec{x} \\ &\wedge Inh(P_{ind}) = \pi_{\overrightarrow{args} \cup \overrightarrow{rets}}(rename_{t^{-1}}(A)) \oplus deps(n) \wedge Inh(Q_{ind}) = \pi_{\overrightarrow{args} \cup \overrightarrow{rets}}(rename_{t^{-1}}(A')) \oplus deps(n)) \\ &\wedge rename_{t^{-1}}(|ind|) \oplus Inh(P) = rename_{t^{-1}}(|ind|) \oplus Inh(P_{ind}) \\ &\wedge rename_{t^{-1}}(|ind|) \oplus Inh(Q) = rename_{t^{-1}}(|ind|) \oplus Inh(Q_{ind}) \wedge wellAnnot_M^n(\vec{U} \cup modif(s'), read(s'), A, s') \end{aligned}$$

  where
  - $t := (\overrightarrow{args} \cup \overrightarrow{rets}, \vec{x} \cup \vec{y}, \vec{U}, modif(s))$
  - $s' := rename_t(s)$
  - $A' := sem_M(A, inl_M^n(\vec{U} \cup modif(s'), s'))$
- *Loop (depth $= n+1$)*

  $$\begin{aligned} &wellAnnot_M^{n+1}(\vec{U}, \vec{V}, A, \textbf{while } (b) \textbf{ inv } I \ \{s\}) \\ \iff &(\exists ind \in \Sigma. A\#|ind| \wedge |ind| \oplus deps(n+1) \oplus Inh(I) = \pi_V(A) \wedge \sigma(ind) \cap modif(s) = \varnothing) \\ &\wedge (\exists ind \in \Sigma. \ A'\#|ind| \wedge |ind| \oplus deps(n+1) \oplus Inh(I) = \pi_V(A') \wedge \sigma(ind) \cap modif(s) = \varnothing) \\ &\wedge wellAnnot_M^n(\vec{U}, \vec{V}, f_b(A), s) \wedge wellAnnot_M^n(\vec{U} \cup modif(s'), \vec{V}, sem_M(f_b(A), s'), \textbf{while } (b) \textbf{ inv } I \ \{s\}) \end{aligned}$$

  where

- $s' := inl^n_M(\vec{U}, s)$
- $A' := sem_M(f_b(A), s')$

- *Loop (depth = 0)*

$$wellAnnot^0_M(\vec{U}, \vec{V}, A, \textbf{while } (b) \textbf{ inv } I \ \{s\}) \Longleftrightarrow (\exists ind \in \Sigma. A\#|ind| \wedge |ind| \oplus deps(0) \oplus Inh(I) = \pi_V(A) \wedge \sigma(ind) \cap modif(s) = \varnothing)$$

- *Sequential composition*

$$\begin{aligned} & wellAnnot^n_M(\vec{U}, \vec{V}, A, s_1 \ ; \ s_2) \\ \Longleftrightarrow & wellAnnot^n_M(\vec{U}, \vec{V}, A, s_1) \wedge wellAnnot^n_M(\vec{U} \cup read(s'_1), \vec{V}, sem_M(A, s'_1), s_2) \qquad (\text{where } s'_1 := inl^n_M(\vec{U}, s_1)) \end{aligned}$$

- *Non-deterministic branching*

$$wellAnnot^n_M(\vec{U}, \vec{V}, A, \textbf{if } (*) \ \{s_1\} \textbf{ else } \{s_2\}) \Longleftrightarrow wellAnnot^n_M(\vec{U}, \vec{V}, A, s_1) \wedge wellAnnot^n_M(\vec{U}, \vec{V}, A, s_2)$$

- *Other cases*

$$wellAnnot^n_M(\vec{U}, \vec{V}, A, s) \Longleftrightarrow \top$$

3 Completeness

3.1 Theorem

We now express the completeness theorem, where $verMethodsB(M)$ expresses that all methods of M verify w.r.t. their annotations, and $annot(M, collectAn^n_M(modif(s), read(s), \{u\}, s))$ is the set of methods M that have been annotated with the quadruples $collectAn^n_M(modif(s), read(s), \{u\}, s)$. The annotation corresponding to $collectAn^n_M(modif(s), read(s), \{u\}, s)$ is the same as the one expressed in the *wellAnnot* function.

Theorem 1. *If*

1. *The program is well-formed:* $wfStmt_M(s) \wedge wfMethods(M)$
2. *The program is well-annotated:* $wellAnnot^n_M(modif(s), read(s), \{u\}, s)$
3. *Loop iterations do not create new variables.*
4. $ver_M(\{u\}, inl^n_M(modif(s), s))$

Then

1. $verB_M(\{deps(n)\}, s)$
2. $verMethodsB(annot(M, collectAn^n_M(modif(s), read(s), \{u\}, s)))$

3.2 Property proved by induction

To prove the completeness theorem, we prove the following stronger property by computational induction on the structure of the inline function:

Definition 6. *The completeness invariant, written* $CompletenessInv^n_M(s)$*, holds if and only if:*
For all set of states A, sets of variable names $\vec{V}$ *and* $\vec{U}$*, if*

1. $wfStmt_M(s) \wedge wfMethods(M)$
2. $read(s) \subseteq \vec{V} \wedge dep \notin \vec{V}$
3. $modif(s) \subseteq \vec{U} \wedge dom(A) \subseteq \vec{U}$
4. $wellAnnot^n_M(\vec{U}, \vec{V}, A, s)$
5. $ver_M(A, inl^n_M(\vec{U}, s))$
6. *Loop iterations do not create new variables*

Then

1. $verB_M(A \oplus deps(n), s)$
2. $\pi_{\vec{V}}(semB_M(A \oplus deps(n), s)) = \pi_{\vec{V}}(sem_M(A, inl^n_M(\vec{U}, s)))$
3. $verMethodsB(annot(M, collectAn^n_M(\vec{U}, \vec{V}, A, s)))$

This property is trivial for statements which do not contain any loops or any method calls, since the inlined program is the same as the original program, and nothing has to be proved for $collectAn^n_M(\vec{U}, \vec{V}, A, s)))$ since it is empty.

4 Induction case: Non-deterministic branching

Lemma 5. *If*

1. $CompletenessInv^n_M(s_1)$
2. $CompletenessInv^n_M(s_2)$

then

$$CompletenessInv^n_M(\textbf{if } (*) \ \{s_1\} \textbf{ else } \{s_2\})$$

Proof. Let $s := (\textbf{if } (*) \ \{s_1\} \textbf{ else } \{s_2\})$, we assume $CompletenessInv^n_M(s_1)$ and $CompletenessInv^n_M(s_2)$.
Let A be a set of states, $\vec{V}$ and $\vec{U}$ sets of variable names.
To prove the invariant, we assume that

1. $wfStmt_M(s) \wedge wfMethods(M)$
2. $read(s) \subseteq \vec{V} \wedge dep \notin \vec{V}$
3. $modif(s) \subseteq \vec{U} \wedge dom(A) \subseteq \vec{U}$
4. $wellAnnot^n_M(\vec{U}, \vec{V}, A, s)$
5. $ver_M(A, inl^n_M(\vec{U}, s))$
6. Loop iterations do not create new variables

We need to prove the following points:

1. $verB_M(A \oplus deps(n), \textbf{if } (*) \ \{s_1\} \textbf{ else } \{s_2\})$
2. $\pi_{\vec{V}}(sem_M(A, inl^n_M(\vec{U}, \textbf{if } (*) \ \{s_1\} \textbf{ else } \{s_2\}))) = \pi_{\vec{V}}(semB_M(A \oplus deps(n), \textbf{if } (*) \ \{s_1\} \textbf{ else } \{s_2\}))$
3. $verMethodsB(annot(M, collectAn^n_M(\vec{U}, \vec{V}, A, \textbf{if } (*) \ \{s_1\} \textbf{ else } \{s_2\})))$

We have $ver_M(A, inl^n_M(\vec{U}, s_1))$. From $CompletenessInv^n_M(s_1)$, we get

$$verB_M(A \oplus deps(n), s_1) \tag{1}$$
$$\pi_{\vec{V}}(sem_M(A, inl^n_M(\vec{U}, s_1))) = \pi_{\vec{V}}(semB_M(A \oplus deps(n), s_1)) \tag{2}$$
$$verMethodsB(annot(M, collectAn^n_M(\vec{U}, \vec{V}, A, s_1))) \tag{3}$$

We have $ver_M(A, inl^n_M(\vec{U}, s_2))$. From $CompletenessInv^n_M(s_2)$, we get

$$verB_M(A \oplus deps(n), s_2) \tag{4}$$
$$\pi_{\vec{V}}(sem_M(A, inl^n_M(\vec{U}, s_2))) = \pi_{\vec{V}}(semB_M(A \oplus deps(n), s_2)) \tag{5}$$
$$verMethodsB(annot(M, collectAn^n_M(\vec{U}, \vec{V}, A, s_2))) \tag{6}$$

4.1 Point 1

$$verB_M(A, s) \iff verB_M(A, s_1) \wedge verB_M(A, s_2)$$

We conclude using Equation 1 and Equation 4.

4.2 Point 2

$$\begin{aligned}
\pi_{\vec{V}}(sem_M(A, inl^n_M(\vec{U}, s))) =& \pi_{\vec{V}}(sem_M(A, \textbf{if } (*) \ \{inl^n_M(\vec{U}, s_1)\} \textbf{ else } \{inl^n_M(\vec{U}, s_2)\})) \\
& \pi_{\vec{V}}(sem_M(A, inl^n_M(\vec{U}, s_1)) \cup sem_M(A, inl^n_M(\vec{U}, s_2))) \\
& \pi_{\vec{V}}(sem_M(A, inl^n_M(\vec{U}, s_1))) \cup \pi_{\vec{V}}(sem_M(A, inl^n_M(\vec{U}, s_2))) \\
& \pi_{\vec{V}}(semB_M(A, s_1)) \cup \pi_{\vec{V}}(semB_M(A, s_2)) \qquad \text{(Equations 2 and 5)} \\
& \pi_{\vec{V}}(semB_M(A, s_1) \cup semB_M(A, s_2)) \\
& \pi_{\vec{V}}(semB_M(A, s))
\end{aligned}$$

4.3 Point 3

Since $collectAn^n_M(\vec{U}, \vec{V}, A, s) = collectAn^n_M(\vec{U}, \vec{V}, A, s_1) \cup collectAn^n_M(\vec{U}, \vec{V}, A, s_2)$, we have

$$\begin{aligned}
& verMethodsB(annot(M, collectAn^n_M(\vec{U}, \vec{V}, A, s))) \\
\iff & verMethodsB(annot(M, collectAn^n_M(\vec{U}, \vec{V}, A, s_1))) \wedge verMethodsB(annot(M, collectAn^n_M(\vec{U}, \vec{V}, A, s_2)))
\end{aligned}$$

which corresponds to Equation 3 and Equation 6.

□

5 Induction case: Sequential composition

Lemma 6. *If*

1. $CompletenessInv^n_M(s_1)$
2. $CompletenessInv^n_M(s_2)$

then

$$CompletenessInv^n_M(s_1 \ ; \ s_2)$$

Proof. Let $s := (s_1 \ ; \ s_2)$, we assume $CompletenessInv^n_M(s_1)$ and $CompletenessInv^n_M(s_2)$.

Let A be a set of states, $\vec{V}$ and $\vec{U}$ sets of variable names.

To prove the invariant, we assume that

1. $wfStmt_M(s) \wedge wfMethods(M)$
2. $read(s) \subseteq \vec{V} \wedge dep \notin \vec{V}$
3. $modif(s) \subseteq \vec{U} \wedge dom(A) \subseteq \vec{U}$
4. $wellAnnot^n_M(\vec{U}, \vec{V}, A, s)$
5. $ver_M(A, inl^n_M(\vec{U}, s))$
6. Loop iterations do not create new variables

We need to prove the following points:

1. $verB_M(A \oplus deps(n), s_1 \ ; \ s_2)$
2. $\pi_{\vec{V}}(sem_M(A, inl^n_M(\vec{U}, s_1 \ ; \ s_2))) = \pi_{\vec{V}}(semB_M(A \oplus deps(n), s_1 \ ; \ s_2))$
3. $verMethodsB(annot(M, collectAn^n_M(\vec{U}, \vec{V}, A, s_1 \ ; \ s_2)))$

We have $ver_M(A, inl^n_M(\vec{U}, s_1))$. From $CompletenessInv^n_M(s_1)$, we get

$$verB_M(A \oplus deps(n), s_1) \tag{7}$$
$$\pi_{\vec{V}}(sem_M(A, inl^n_M(\vec{U}, s_1))) = \pi_{\vec{V}}(semB_M(A \oplus deps(n), s_1)) \tag{8}$$
$$verMethodsB(annot(M, collectAn^n_M(\vec{U}, \vec{V}, A, s_1))) \tag{9}$$

Let $s'_1 := inl^n_M(\vec{U}, s_1)$, and $A' := sem_M(A, s'_1)$. From $ver_M(A, s'_1 \ ; \ inl^n_M(\vec{U} \cup read(s'_1), s_2))$ we get $ver_M(A', inl^n_M(\vec{U}, s_2))$. From $CompletenessInv^n_M(s_2)$, we get

$$verB_M(A' \oplus deps(n), s_2) \tag{10}$$
$$\pi_{\vec{V}}(sem_M(A', inl^n_M(\vec{U} \cup read(s'_1), s_2))) = \pi_{\vec{V}}(semB_M(A' \oplus deps(n), s_2)) \tag{11}$$
$$verMethodsB(annot(M, collectAn^n_M(\vec{U} \cup read(s'_1), \vec{V}, A', s_2))) \tag{12}$$

5.1 Point 1

$$\begin{aligned}
verB_M(A, s_1 \ ; \ s_2) \iff & verB_M(A, s_1) \wedge verB_M(sem_M(A, s_1), s_2) \\
\iff & verB_M(A, s_1) \wedge verB_M(\pi_{\vec{V}}(sem_M(A, s_1)), s_2) && \text{(Since } read(s_2) \subseteq \vec{V}\text{)} \\
\iff & verB_M(A, s_1) \wedge verB_M(\pi_{\vec{V}}(sem_M(A, inl^n_M(\vec{U}, s_1))), s_2) && \text{(Using Equation 8)} \\
\iff & verB_M(A, s_1) \wedge verB_M(\pi_{\vec{V}}(A', s_2)) \\
\iff & verB_M(A, s_1) \wedge verB_M(A', s_2) && \text{(Since } read(s_2) \subseteq \vec{V}\text{)}
\end{aligned}$$

We conclude using Equation 7 and Equation 10.

5.2 Point 2

$$\begin{aligned}
\pi_{\vec{V}}(sem_M(A, inl^n_M(\vec{U}, s_1 \ ; \ s_2))) = & \pi_{\vec{V}}(sem_M(A, inl^n_M(\vec{U}, s_1) \ ; \ inl^n_M(\vec{U} \cup read(s'_1), s_2))) \\
= & \pi_{\vec{V}}(sem_M(sem_M(A, inl^n_M(\vec{U}, s_1)), inl^n_M(\vec{U} \cup read(s'_1), s_2))) \\
= & \pi_{\vec{V}}(sem_M(A', inl^n_M(\vec{U} \cup read(s'_1), s_2))) \\
= & \pi_{\vec{V}}(semB_M(A', s_2)) && \text{(Using Equation 11)} \\
= & \pi_{\vec{V}}(semB_M(\pi_{\vec{V}}(A'), s_2)) && \text{(Since } read(s_2) \subseteq \vec{V}\text{)} \\
= & \pi_{\vec{V}}(semB_M(\pi_{\vec{V}}(semB_M(A, s_1)), s_2)) && \text{(Using Equation 8)} \\
= & \pi_{\vec{V}}(semB_M(semB_M(A, s_1), s_2)) && \text{(Since } read(s_2) \subseteq \vec{V}\text{)} \\
= & \pi_{\vec{V}}(semB_M(A, s_1 \ ; \ s_2))
\end{aligned}$$

5.3 Point 3

Since $collectAn^n_M(\vec{U}, \vec{V}, A, s_1 \,;\, s_2) = collectAn^n_M(\vec{U}, \vec{V}, A, s_1) \cup collectAn^n_M(\vec{U} \cup read(s_1'), \vec{V}, A', s_2)$, we have

$$\begin{aligned} & verMethodsB(annot(M, collectAn^n_M(\vec{U}, \vec{V}, A, s_1 \,;\, s_2))) \\ \iff & verMethodsB(annot(M, collectAn^n_M(\vec{U}, \vec{V}, A, s_1))) \wedge verMethodsB(annot(M, collectAn^n_M(\vec{U} \cup read(s_1'), \vec{V}, A, s_2))) \end{aligned}$$

which corresponds to Equation 9 and Equation 12.

□

6 Induction case: Method ($depth = n = 0$)

Lemma 7.

$$CompletenessInv^0_M(\overrightarrow{y} := m(\overrightarrow{x}))$$

Proof. Let A be a set of states, $\vec{V}$ and $\vec{U}$ sets of variable names.
To prove the invariant, we assume that

1. $wfStmt_M(\overrightarrow{y} := m(\overrightarrow{x})) \wedge wfMethods(M)$
2. $read(\overrightarrow{y} := m(\overrightarrow{x})) \subseteq \vec{V} \wedge dep \notin \vec{V}$
3. $modif(\overrightarrow{y} := m(\overrightarrow{x})) \subseteq \vec{U} \wedge dom(A) \subseteq \vec{U}$
4. $wellAnnot^0_M(\vec{U}, \vec{V}, A, \overrightarrow{y} := m(\overrightarrow{x}))$
5. $ver_M(A, inl^0_M(\vec{U}, \overrightarrow{y} := m(\overrightarrow{x})))$
6. Loop iterations do not create new variables

We need to prove the following points:

1. $verB_M(A \oplus deps(0), \overrightarrow{y} := m(\overrightarrow{x}))$
   True by definition.
2. $\pi_{\vec{V}}(sem_M(A, inl^0_M(\vec{U}, \overrightarrow{y} := m(\overrightarrow{x})))) = \pi_{\vec{V}}(semB_M(A \oplus deps(0), \overrightarrow{y} := m(\overrightarrow{x})))$
   - $inl^0_M(\vec{U}, \overrightarrow{y} := m(\overrightarrow{x})) = \textbf{assume } \bot$
   - $semB_M(A, \overrightarrow{y} := m(\overrightarrow{x})) = \varnothing$
   - $\pi_{\vec{V}}(sem_M(A, inl^0_M(\vec{U}, \overrightarrow{y} := m(\overrightarrow{x})))) = \pi_{\vec{V}}(\varnothing) = \pi_{\vec{V}}(semB_M(A \oplus deps(0), \overrightarrow{y} := m(\overrightarrow{x})))$
3. $verMethodsB(annot(M, collectAn^0_M(\vec{U}, \vec{V}, A, \overrightarrow{y} := m(\overrightarrow{x}))))$
   Directly follows from $collectAn^0_M(\vec{U}, \vec{V}, A, \overrightarrow{y} := m(\overrightarrow{x})) = \varnothing$

□

7 Induction case: Loop ($depth = n = 0$)

Lemma 8.

$$CompletenessInv^0_M(\textbf{while } (b) \textbf{ inv } I \ \{s\})$$

Proof. Let A be a set of states, $\vec{V}$ and $\vec{U}$ sets of variable names.
To prove the invariant, we assume that

1. $wfStmt_M(\textbf{while } (b) \textbf{ inv } I \ \{s\}) \wedge wfMethods(M)$
2. $read(\textbf{while } (b) \textbf{ inv } I \ \{s\}) \subseteq \vec{V} \wedge dep \notin \vec{V}$
3. $modif(\textbf{while } (b) \textbf{ inv } I \ \{s\}) \subseteq \vec{U} \wedge dom(A) \subseteq \vec{U}$
4. $wellAnnot^0_M(\vec{U}, \vec{V}, A, \textbf{while } (b) \textbf{ inv } I \ \{s\})$
5. $ver_M(A, inl^0_M(\vec{U}, \textbf{while } (b) \textbf{ inv } I \ \{s\}))$
6. Loop iterations do not create new variables

Let $\varphi \in A$ and $\overrightarrow{l} := modif(s) \cap \sigma(\varphi)$. We have

$$\begin{aligned} & sem_M(\{\varphi\} \oplus deps(0), \textbf{exhale } I \,;\, \textbf{havoc } \overrightarrow{l} \,;\, \textbf{assign } 0 \leq dep \leq 0 \,;\, \textbf{inhale } I) \\ =& sem_M(\{|varphi|\} \oplus deps(0), \textbf{havoc } \overrightarrow{l} \,;\, \textbf{assign } 0 \leq dep \leq 0 \,;\, \textbf{inhale } I) && \text{(Well-annotated hypothesis)} \\ =& sem_M(\{\overline{h}_{\overrightarrow{l}}(|varphi|)\} \oplus h(\overrightarrow{l}) \oplus deps(0), \textbf{assign } 0 \leq dep \leq 0 \,;\, \textbf{inhale } I) \\ =& sem_M(\{\overline{h}_{\overrightarrow{l}}(|varphi|)\} \oplus h(\overrightarrow{l}) \oplus deps(0), \textbf{inhale } I) \\ =& \{\overline{h}_{\overrightarrow{l}}(|varphi|)\} \oplus h(\overrightarrow{l}) \oplus deps(0) \oplus Inh(I) \\ =& \{\overline{h}_{\overrightarrow{l}}(|varphi|)\} \oplus h(\overrightarrow{l}) \oplus deps(0) \oplus \pi_{\vec{V}}(A) && \text{(Well-annotated hypothesis)} \end{aligned}$$

Thus, since $\overrightarrow{l} \subseteq modif(s) \subseteq read(s) \subseteq \vec{V}$,

$$\pi_{\vec{V}}(\{\varphi\}) \subseteq \pi_{\vec{V}}(sem_M(\{\varphi\} \oplus deps(0), \textbf{exhale } I \,;\, \textbf{havoc } \overrightarrow{l} \,;\, \textbf{assign } 0 \leq dep \leq 0 \,;\, \textbf{inhale } I)) \subseteq \pi_{\vec{V}}(A)$$

Therefore

$$\pi_{\vec{V}}(sem_M(A \oplus deps(0), \textbf{exhale } I \textbf{ ; havoc } \vec{l} \textbf{ ; assign } 0 \leq dep \leq 0 \textbf{ ; inhale } I)) = A \tag{13}$$

We need to prove the following points:

1. $verB_M(A, \textbf{while } (b) \textbf{ inv } I \ \{s\})$
   - $verB_M(\{|\varphi|\}, \textbf{havoc } \vec{l} \textbf{ ; assign } 1 \leq dep \leq 0 \textbf{ ; inhale } I \textbf{ ; assume } b \textbf{ ; } dep \mathrel{-}= 1 \textbf{ ; } s \textbf{ ; exhale } I)$
     since $semB_M(\{|\varphi|\}, \textbf{havoc } \vec{l} \textbf{ ; assign } 1 \leq dep \leq 0) = \varnothing$
   - $ver_M(\{\varphi\}, \textbf{exhale } I \textbf{ ; havoc } \vec{l} \textbf{ ; assign } 0 \leq dep \leq 0 \textbf{ ; inhale } I \textbf{ ; assume } \neg b \textbf{ ; } dep \leftarrow 0)$
     using Equation 13
2. $\pi_{\vec{V}}(sem_M(A, inl^0_M(\textbf{while } (b) \textbf{ inv } I \ \{s\}))) = \pi_{\vec{V}}(semB_M(A, \textbf{while } (b) \textbf{ inv } I \ \{s\}))$

$$\begin{aligned}\pi_{\vec{V}}(sem_M(A, inl^0_M(\textbf{while } (b) \textbf{ inv } I \ \{s\}))) &= \pi_{\vec{V}}(sem_M(A, \textbf{assume } \neg b)) \\ &= \pi_{\vec{V}}(f_{\neg b}(A)) \\ &= \pi_{\vec{V}}(semB_M(A, \textbf{while } (b) \textbf{ inv } I \ \{s\})) \qquad \text{(Using Equation 13)}\end{aligned}$$

3. $verMethodsB(annot(M, collectAn^0_M(\vec{U}, \vec{V}, A, \textbf{while } (b) \textbf{ inv } I \ \{s\})))$
   Directly follows from $collectAn^0_M(\vec{U}, \vec{V}, A, \textbf{while } (b) \textbf{ inv } I \ \{s\}) = \varnothing$

□

8 Induction case: Method ($depth = n + 1$)

Lemma 9. *If*

1. *The method name m corresponds to the method $(\overrightarrow{args}, \overrightarrow{rets}, P, Q, s)$ in the set of methods M.*
2. $CompletenessInv^n_M(rename_{(\overrightarrow{args} \cup \overrightarrow{rets}, \vec{x} \cup \vec{y}, \vec{U}, modif(s))}(s))$

Then

$$CompletenessInv^{n+1}_M(\overrightarrow{y} := m(\overrightarrow{x}))$$

Proof. Let $t := (\overrightarrow{args} \cup \overrightarrow{rets}, \vec{x} \cup \vec{y}, \vec{U}, modif(s))$ be a renaming configuration.
Let $s' := rename_t(s)$. We have

$$inl^{n+1}_M(\overrightarrow{y} := m(\overrightarrow{x})) = inl^n_M(\vec{U} \cup modif(s'), s')$$

Let us assume $CompletenessInv^n_M(s')$.
Let A be a set of states, $\vec{V}$ and $\vec{U}$ sets of variable names.
To prove the invariant, we assume that

1. $wfStmt_M(\overrightarrow{y} := m(\overrightarrow{x})) \wedge wfMethods(M)$
2. $read(s) \subseteq \vec{V} \wedge dep \notin \vec{V}$
3. $modif(s) \subseteq \vec{U} \wedge dom(A) \subseteq \vec{U}$
4. $wellAnnot^n_M(\vec{U}, \vec{V}, A, \overrightarrow{y} := m(\overrightarrow{x}))$
5. $ver_M(A, inl^n_M(\vec{U}, \overrightarrow{y} := m(\overrightarrow{x})))$
6. Loop iterations do not create new variables

Let $P' := rename_{(\overrightarrow{args} \cup \overrightarrow{rets}, \vec{x} \cup \vec{y}, \varnothing, \varnothing)}(P)$, and $Q' := rename_{(\overrightarrow{args} \cup \overrightarrow{rets}, \vec{x} \cup \vec{y}, \varnothing, \varnothing)}(Q)$.
Since $read(P) \subseteq \vec{x}$, we have $P' = rename_t(P)$. Similarly, since $read(Q) \subseteq \vec{x} \cup \vec{y}$, we have $Q' = rename_t(Q)$.
We need to prove the following points:

1. $verB_M(A \oplus deps(n+1), dep \mathrel{-}= 1 \textbf{ ; exhale } P' \textbf{ ; havoc } \vec{y} \textbf{ ; inhale } Q \textbf{ ; } dep \mathrel{+}= 1)$
2. $\pi_{\vec{V}}(semB_M(A \oplus deps(n+1), dep \mathrel{-}= 1 \textbf{ ; } \vec{y} := m(\vec{x}) \textbf{ ; } dep \mathrel{+}= 1))$
   $= \pi_{\vec{V}}(sem_M(A, inl^{n+1}_M(\vec{U}, \overrightarrow{y} := m(\overrightarrow{x}))))$
3. $verMethodsB(annot(M, collectAn^{n+1}_M(\vec{U}, \vec{V}, A, \overrightarrow{y} := m(\overrightarrow{x}))))$

From $CompletenessInv^n_M(s')$, we get

$$verB_M(A \oplus deps(n), s') \tag{14}$$

$$\pi_{read(s')}(semB_M(A \oplus deps(n), s')) = \pi_{read(s')}(sem_M(A, inl^n_M(\vec{U} \cup modif(s'), s'))) \tag{15}$$

$$verMethodsB(annot(M, collectAn^n_M(\vec{U} \cup modif(s'), read(s'), A, s'))) \tag{16}$$

Let $\vec{ar} := \overrightarrow{args} \cup \overrightarrow{rets}$

From hypothesis 4 ($wellAnnot^n_M(\vec{U}, \vec{V}, A, \overrightarrow{y} := m(\overrightarrow{x}))$), there exists ind, P_{ind}, Q_{ind} such that

$$A\#|ind| \tag{17}$$
$$sem_M(A, inl^n_M(s'))\#|ind| \tag{18}$$
$$\sigma(|ind|) \subseteq \vec{x} \tag{19}$$
$$rename_{t^{-1}}(|ind|) \oplus Inh(P) = rename_{t^{-1}}(|ind|) \oplus Inh(P_{ind}) \tag{20}$$
$$rename_{t^{-1}}(|ind|) \oplus Inh(Q) = rename_{t^{-1}}(|ind|) \oplus Inh(Q_{ind}) \tag{21}$$

and

$$Inh(P_{ind}) = \pi_{\overrightarrow{args} \cup \overrightarrow{rets}}(rename_{t^{-1}}(A)) \oplus deps(n) \wedge Inh(Q_{ind}) = \pi_{\overrightarrow{args} \cup \overrightarrow{rets}}(rename_{t^{-1}}(sem_M(A, inl^n_M(\vec{U} \cup modif(s'), s')))) \oplus deps(n)$$

Therefore we have

$$Inh(rename_t(P_{ind})) = \pi_{\vec{x} \cup \vec{y}}(A) \oplus deps(n) \tag{22}$$
$$Inh(rename_t(Q_{ind})) = \pi_{\vec{x} \cup \vec{y}}(sem_M(A, inl^n_M(\vec{U} \cup modif(s'), s'))) \oplus deps(n) \tag{23}$$

8.1 Proving point 1 and 2

Let $\varphi \in A$. We have

$$\begin{aligned}
&semB_M(\{\varphi\} \oplus deps(n+1), dep \mathrel{-}= 1 \,;\, \textbf{exhale}\ P' \,;\, \textbf{havoc}\ \vec{y} \,;\, \textbf{inhale}\ Q' \,;\, dep \mathrel{+}= 1) \\
=&semB_M(\{\varphi\} \oplus deps(n), \textbf{exhale}\ P' \,;\, \textbf{havoc}\ \vec{y} \,;\, \textbf{inhale}\ Q' \,;\, dep \mathrel{+}= 1) \\
=&semB_M(\{\varphi\} \oplus deps(n), \textbf{exhale}\ rename_t(P) \,;\, \textbf{havoc}\ \vec{y} \,;\, \textbf{inhale}\ Q' \,;\, dep \mathrel{+}= 1) \\
=&semB_M(\{\varphi\} \oplus deps(n), \textbf{exhale}\ rename_t(P_{ind}) \,;\, \textbf{havoc}\ \vec{y} \,;\, \textbf{inhale}\ Q' \,;\, dep \mathrel{+}= 1) && \text{(Using Equations 17 and 19)} \\
=&semB_M(\{|\varphi|\} \oplus deps(n), \textbf{havoc}\ \vec{y} \,;\, \textbf{inhale}\ Q' \,;\, dep \mathrel{+}= 1) && \text{(Using Equation 22)} \\
=&semB_M(\{\overline{h}_{\vec{y}}(|\varphi|)\} \oplus h(y) \oplus deps(n), \textbf{inhale}\ Q' \,;\, dep \mathrel{+}= 1) \\
=&semB_M(\{\overline{h}_{\vec{y}}(|\varphi|)\} \oplus h(y) \oplus deps(n), \textbf{inhale}\ rename_t(Q) \,;\, dep \mathrel{+}= 1) \\
=&semB_M(\{\overline{h}_{\vec{y}}(|\varphi|)\} \oplus h(y) \oplus deps(n) \oplus rename_t(Inh(Q)), dep \mathrel{+}= 1) \\
=&\{\overline{h}_{\vec{y}}(|\varphi|)\} \oplus h(y) \oplus deps(n+1) \oplus rename_t(Inh(Q)) \\
=&\{\overline{h}_{\vec{y}}(|\varphi|)\} \oplus h(y) \oplus deps(n+1) \oplus rename_t(Inh(Q_{ind})) && \text{(Using Equations 18, 20, 21, and } \vec{x} \cap \vec{y} = \varnothing) \\
=&\{\overline{h}_{\vec{y}}(|\varphi|)\} \oplus h(y) \oplus deps(n+1) \oplus \pi_{\vec{x} \cup \vec{y}}(sem_M(A, inl^n_M(\vec{U} \cup modif(s'), s'))) && \text{(Using Equation 23)} \\
=&\{\overline{h}_{\vec{y}}(|\varphi|)\} \oplus \pi_{\vec{x} \cup \vec{y}}(sem_M(A, inl^n_M(\vec{U} \cup modif(s'), s'))) \oplus deps(n+1)
\end{aligned}$$

Therefore

$$\begin{aligned}
&\pi_{\vec{V}}(sem_M(A \oplus deps(n+1), dep \mathrel{-}= 1 \,;\, \vec{y} := m(\vec{x}) \,;\, dep \mathrel{+}= 1)) \\
=&\bigcup_{\varphi \in A}(\pi_{\vec{V}}(\{\overline{h}_{\vec{y}}(|\varphi|)\} \oplus \pi_{\vec{x} \cup \vec{y}}(sem_M(A, inl^n_M(\vec{U} \cup modif(s'), s'))))) \\
=&\bigcup_{\varphi \in A}(\{\pi_{\vec{V} - \vec{y}}(|\varphi|)\}) \oplus \pi_{\vec{x} \cup \vec{y}}(sem_M(A, inl^n_M(\vec{U} \cup modif(s'), s'))) \\
=&\pi_{\vec{V} - \vec{y}}(|A|) \oplus \pi_{\vec{x} \cup \vec{y}}(sem_M(A, inl^n_M(\vec{U} \cup modif(s'), s'))) \\
=&\pi_{\vec{V}}(sem_M(A, inl^n_M(\vec{U} \cup modif(s'), s'))) && \text{(Since } \vec{V} \cap modif(inl^n_M(\vec{U} \cup modif(s'), s')) \subseteq \vec{y}) \\
=&\pi_{\vec{V}}(sem_M(A, inl^{n+1}_M(\vec{U}, \overrightarrow{y} := m(\overrightarrow{x}))))
\end{aligned}$$

8.2 Proving point 3

Since $collectAn^{n+1}_M(\vec{U}, \vec{V}, A, \overrightarrow{y} := m(\overrightarrow{x})) = \{(m, ind, P_{ind}, Q_{ind})\} \cup collectAn^n_M(\vec{U} \cup modif(s'), read(s'), A, s')$, we have

$verMethodsB(annot(M, collectAn^{n+1}_M(\vec{U}, \vec{V}, A, \overrightarrow{y} := m(\overrightarrow{x})))) \iff$
$verB_M(h(args) \oplus h(rets), \textbf{inhale}\ P_{ind} \,;\, s \,;\, \textbf{exhale}\ Q_{ind}) \wedge verMethodsB(annot(M, collectAn^n_M(\vec{U} \cup modif(s'), read(s'), A, s')))$

The first part is given by Equation 14, thus we only need to prove the second part.

By inverting the renaming (which is well-formed), we have

$$
\begin{aligned}
& verB_M(h(args) \oplus h(rets), \textbf{inhale } P_{ind} \textbf{ ; } s \textbf{ ; exhale } Q_{ind}) \\
\iff & verB_M(h(rename_t(\overrightarrow{args})) \oplus h(rename_t(\overrightarrow{rets})), \textbf{inhale } rename_t(P_{ind}) \textbf{ ; } rename_t(s) \textbf{ ; exhale } rename_t(Q_{ind})) \\
\iff & verB_M(h(\vec{x}) \oplus h(\vec{y}), \textbf{inhale } rename_t(P_{ind}) \textbf{ ; } s' \textbf{ ; exhale } rename_t(Q_{ind})) \\
\iff & verB_M(h(\vec{x}) \oplus h(\vec{y}) \oplus rename_t(Inh(P_{ind}))), s' \textbf{ ; exhale } rename_t(Q_{ind})) \\
\iff & verB_M(h(\vec{x}) \oplus h(\vec{y}) \oplus \pi_{\vec{x} \cup \vec{y}}(A) \oplus deps(n), s' \textbf{ ; exhale } rename_t(Q_{ind})) && \text{(Using Equation 22)} \\
\iff & verB_M(\pi_{\vec{x} \cup \vec{y}}(A) \oplus deps(n), s' \textbf{ ; exhale } rename_t(Q_{ind})) \\
\iff & verB_M(\pi_{\vec{x} \cup \vec{y}}(A) \oplus deps(n), s') \wedge semB_M(\pi_{\vec{x} \cup \vec{y}}(A) \oplus deps(n), s') >> rename_t(Inh(Q_{ind}))) && (24)
\end{aligned}
$$

Since $\sigma(A) \cap read(s') \subseteq \vec{x} \cup \vec{y}$ (renaming property), we have

$$verB_M(\pi_{\vec{x} \cup \vec{y}}(A) \oplus deps(n), s') \iff verB_M(A \oplus deps(n), s')$$

given by Equation 14, which concludes the left part of Equation 24.

Moreover

$$
\begin{aligned}
& \pi_{read(s')}(semB_M(\pi_{\vec{x} \cup \vec{y}}(A) \oplus deps(n), s')) \\
= & \pi_{\vec{x} \cup \vec{y}}(semB_M(A \oplus deps(n), s')) && \text{(Since } \vec{x} \cup \vec{y} \subseteq read(s')) \\
= & \pi_{\vec{x} \cup \vec{y}}(\pi_{read(s')}(semB_M(A \oplus deps(n), s'))) && \text{(Since } \vec{x} \cup \vec{y} \subseteq read(s')) \\
= & \pi_{\vec{x} \cup \vec{y}}(\pi_{read(s')}(sem_M(A, inl^n_M(\vec{U} \cup modif(s'), s')))) && \text{(Using Equation 15)} \\
= & \pi_{read(s')}(\pi_{\vec{x} \cup \vec{y}}(sem_M(A, inl^n_M(\vec{U} \cup modif(s'), s')))) \\
= & \pi_{read(s')}(rename_t(Inh(Q_{ind}))) && \text{(Using Equation 23)} \\
= & rename_t(Inh(Q_{ind}))
\end{aligned}
$$

Therefore

$$semB_M(\pi_{\vec{x} \cup \vec{y}}(A) \oplus deps(n), s') >> \pi_{read(s')}(semB_M(\pi_{\vec{x} \cup \vec{y}}(A) \oplus deps(n), s')) = rename_t(Inh(Q_{ind}))$$

which concludes Equation 24.

□

9 Induction case ($depth = n + 1$)

Lemma 10. *If*

1. $CompletenessInv^n_M(s)$
2. $CompletenessInv^n_M(\textbf{while } (b) \textbf{ inv } I \ \{s\})$

then

$$CompletenessInv^{n+1}_M(\textbf{while } (b) \textbf{ inv } I \ \{s\})$$

Proof. Let $w := \textbf{while } (b) \textbf{ inv } I \ \{s\}$, we assume $CompletenessInv^n_M(s)$ and $CompletenessInv^n_M(w)$.

Let $\varphi \in A$, $\vec{l} := modif(s) \cap \sigma(\varphi)$, $s' := inl^n_M(\vec{U}, s)$, and $\overrightarrow{U'} := \vec{U} \cup read(s')$. To prove the invariant, we need to prove the following points:

1. $verB_M(\{|\varphi|\} \oplus deps(n+1), \textbf{havoc } \vec{l} \textbf{ ; assign } 1 \leq dep \leq n+1 \textbf{ ; inhale } I \textbf{ ; assume } b \textbf{ ; } dep -= 1 \textbf{ ; } s \textbf{ ; exhale } I)$
2. $ver_M(\{\varphi\} \oplus deps(n+1), \textbf{exhale } I \textbf{ ; havoc } \vec{l} \textbf{ ; assign } 0 \leq dep \leq n+1 \textbf{ ; inhale } I \textbf{ ; assume } \neg b \textbf{ ; } dep \leftarrow n+1)$
3. $\pi_{\vec{V}}(sem_M(A, inl^{n+1}_M(\overrightarrow{U'}, \textbf{while } (b) \textbf{ inv } I \ \{s\}))) = \pi_{\vec{V}}(semB_M(A \oplus deps(n+1), \textbf{while } (b) \textbf{ inv } I \ \{s\}))$
4. $verMethodsB(annot(M, collectAn^{n+1}_M(\vec{U}, \vec{V}, A, \textbf{while } (b) \textbf{ inv } I \ \{s\})))$

Points 1 and 2 combined yield $verB_M(A \oplus deps(n), \textbf{while } (b) \textbf{ inv } I \ \{s\})$.

By definition, we have

$$inl^{n+1}_M(\vec{U}, \textbf{while } (b) \textbf{ inv } I \ \{s\}) = \textbf{if } (*) \ \{\textbf{assume } b \textbf{ ; } inl^n_M(\vec{U}, s) \textbf{ ; } inl^n_M(\overrightarrow{U'}, \textbf{while } (b) \textbf{ inv } I \ \{s\})\} \textbf{ else } \{\textbf{assume } \neg b\}$$

We thus have $ver_M(A, \textbf{assume } b \textbf{ ; } inl^n_M(\vec{U}, s) \textbf{ ; } inl^n_M(\overrightarrow{U'}, w))$, which gives us $ver_M(f_b(A), inl^n_M(\vec{U}, s))$.
From $CompletenessInv^n_M(s)$, we get

$$
\begin{aligned}
& verB_M(f_b(A) \oplus deps(n), s) && (25) \\
\wedge & \pi_{\vec{V}}(sem_M(f_b(A), inl^n_M(\vec{U}, s))) = \pi_{\vec{V}}(semB_M(f_b(A) \oplus deps(n), s)) && (26) \\
\wedge & verMethodsB(annot(M, collectAn^n_M(\vec{U}, \vec{V}, A, s))) && (27)
\end{aligned}
$$

Let $s' := inl^n_M(\vec{U}, s)$, and $A' := sem_M(f_b(A), s')$. Using $CompletenessInv^n_M(\overrightarrow{U'}, w)$, we get

$$verB_M(A' \oplus deps(n), w) \tag{28}$$

$$\wedge \pi_{\vec{V}}(sem_M(A', inl^n_M(\overrightarrow{U'}, w))) = \pi_{\vec{V}}(semB_M(A' \oplus deps(n), w)) \tag{29}$$

$$\wedge verMethodsB(annot(M, collectAn^n_M(\overrightarrow{U'}, \vec{V}, A', w))) \tag{30}$$

9.1 Proving point 4

By definition, we have
$collectAn^{n+1}_M(\vec{U}, \vec{V}, A', \textbf{while}\ (b)\ \textbf{inv}\ I\ \{s\}) = collectAn^n_M(\vec{U}, \vec{V}, A, s) \cup collectAn^n_M(\overrightarrow{U'}, \vec{V}, A', \textbf{while}\ (b)\ \textbf{inv}\ I\ \{s\})$
Combined with Equation 27 and Equation 30, we get **point 3**.

9.2 Dividing point 1

We have

$$verB_M(\{|\varphi|\} \oplus deps(n+1), \textbf{havoc}\ \overrightarrow{l}\ ;\ \textbf{assign}\ 1 \leq dep \leq n+1\ ;\ \textbf{inhale}\ I\ ;\ \textbf{assume}\ b\ ;\ dep -= 1\ ;\ s\ ;\ \textbf{exhale}\ I)$$

$$\Longleftrightarrow verB_M(\{|\varphi|\} \oplus deps(n+1), \textbf{havoc}\ \overrightarrow{l}\ ;\ \textbf{assign}\ dep = n+1\ ;\ \textbf{inhale}\ I\ ;\ \textbf{assume}\ b\ ;\ dep -= 1\ ;\ s\ ;\ \textbf{exhale}\ I) \tag{Point 1.1}$$

$$\wedge verB_M(\{|\varphi|\} \oplus deps(n+1), \textbf{havoc}\ \overrightarrow{l}\ ;\ \textbf{assign}\ 1 \leq dep \leq n\ ;\ \textbf{inhale}\ I\ ;\ \textbf{assume}\ b\ ;\ dep -= 1\ ;\ s\ ;\ \textbf{exhale}\ I) \tag{Point 1.2}$$

9.3 Proving point 1.1

We have

$$\begin{aligned}
&semB_M(\{|\varphi|\} \oplus deps(n+1), \textbf{havoc}\ \overrightarrow{l}\ ;\ \textbf{assign}\ dep = n+1\ ;\ \textbf{inhale}\ I) \\
=&\{\overline{h}_{\overrightarrow{l}}(|\varphi|)\} \oplus h(\overrightarrow{l}) \oplus deps(n+1) \oplus Inh(I) && \text{(Normal semantics)} \\
=&\{\overline{h}_{\overrightarrow{l}}(|\varphi|)\} \oplus h(\overrightarrow{l}) \oplus deps(n+1) \oplus \pi_{\vec{V}}(A) && \text{(Well-annotated hypothesis)}
\end{aligned}$$

It follows that

$$\pi_{\vec{V}}(semB_M(\{|\varphi|\} \oplus deps(n+1), \textbf{havoc}\ \overrightarrow{l}\ ;\ \textbf{assign}\ dep = n+1\ ;\ \textbf{inhale}\ I)) \subseteq \pi_{\vec{V}}(A) \oplus deps(n+1)$$

Then

$$\pi_{\vec{V}}(semB_M(\{|\varphi|\}, \textbf{havoc}\ \overrightarrow{l}\ ;\ \textbf{assign}\ dep = n+1\ ;\ \textbf{inhale}\ I\ ;\ \textbf{assume}\ b\ ;\ dep -= 1)) \subseteq \pi_{\vec{V}}(f_b(A)) \oplus deps(n)$$

From Equation 25 and since $read(s) \subseteq \vec{V}$, we get:

$$verB_M(\{|\varphi|\} \oplus deps(n+1), \textbf{havoc}\ \overrightarrow{l}\ ;\ \textbf{assign}\ dep = n+1\ ;\ \textbf{inhale}\ I\ ;\ \textbf{assume}\ b\ ;\ dep -= 1\ ;\ s)$$

and

$$\begin{aligned}
&semB_M(\{|\varphi|\} \oplus deps(n+1), \textbf{havoc}\ \overrightarrow{l}\ ;\ \textbf{assign}\ dep = n+1\ ;\ \textbf{inhale}\ I\ ;\ \textbf{assume}\ b\ ;\ dep -= 1\ ;\ s) \\
>>&\pi_{\vec{V}}(semB_M(\{|\varphi|\} \oplus deps(n+1), \textbf{havoc}\ \overrightarrow{l}\ ;\ \textbf{assign}\ dep = n+1\ ;\ \textbf{inhale}\ I\ ;\ \textbf{assume}\ b\ ;\ dep -= 1\ ;\ s)) \\
=&\pi_{\vec{V}}(semB_M(semB_M(\{|\varphi|\} \oplus deps(n+1), \textbf{havoc}\ \overrightarrow{l}\ ;\ \textbf{assign}\ dep = n+1\ ;\ \textbf{inhale}\ I\ ;\ \textbf{assume}\ b\ ;\ dep -= 1), s)) \\
=&semB_M(\pi_{\vec{V}}(semB_M(\{|\varphi|\} \oplus deps(n+1)), \textbf{havoc}\ \overrightarrow{l}\ ;\ \textbf{assign}\ dep = n+1\ ;\ \textbf{inhale}\ I\ ;\ \textbf{assume}\ b\ ;\ dep -= 1), s) \\
& && \text{(Since } read(s) \subseteq \vec{V}\text{)} \\
>>&semB_M(\pi_{\vec{V}}(f_b(A) \oplus deps(n)), s) \\
=&\pi_{\vec{V}}(semB_M(f_b(A) \oplus deps(n), s)) \\
=&\pi_{\vec{V}}(sem_M(f_b(A, inl^n_M(\vec{U}, s)))) && \text{(Using Equation 26)} \\
=&Inh(I) && \text{(Well-formed annotated hypothesis)}
\end{aligned}$$

This concludes the case, by definition of the verification of **exhale** statements. Therefore, it remains to prove **point 1.2** to have **point 1**.

9.4 Dividing point 2

For all $\varphi \in A$:

$$ver_M(\{\varphi\} \oplus deps(n+1), \textbf{exhale}\ I\ ;\ \textbf{havoc}\ \overrightarrow{l}\ ;\ \textbf{assign}\ 0 \leq dep \leq n+1\ ;\ \textbf{inhale}\ I\ ;\ \textbf{assume}\ \neg b\ ;\ dep \leftarrow n+1)$$

$$\Longleftrightarrow ver_M(\{\varphi\} \oplus deps(n+1), \textbf{exhale}\ I\ ;\ \textbf{havoc}\ \overrightarrow{l}\ ;\ \textbf{assign}\ dep = n+1\ ;\ \textbf{inhale}\ I\ ;\ \textbf{assume}\ \neg b\ ;\ dep \leftarrow n+1) \tag{Point 2.1}$$

$$\wedge ver_M(\{\varphi\} \oplus deps(n+1), \textbf{exhale}\ I\ ;\ \textbf{havoc}\ \overrightarrow{l}\ ;\ \textbf{assign}\ 0 \leq dep \leq n\ ;\ \textbf{inhale}\ I\ ;\ \textbf{assume}\ \neg b\ ;\ dep \leftarrow n+1) \tag{Point 2.2}$$

9.5 Dividing point 3

We want to show

$$\pi_{\vec{V}}(sem_M(A, inl_M^{n+1}(\vec{U}, \textbf{while } (b) \textbf{ inv } I \ \{s\}))) = \pi_{\vec{V}}(semB_M(A \oplus deps(n+1), \textbf{while } (b) \textbf{ inv } I \ \{s\}))$$

We have

$$\begin{aligned} sem_M(A, inl_M^{n+1}(\vec{U}, \textbf{while } (b) \textbf{ inv } I \ \{s\})) =& sem_M(A, \textbf{if } (*) \ \{\textbf{assume } b \ ; \ inl_M^n(\vec{U}, s) \ ; \ inl_M^n(\overrightarrow{U'}, w)\} \textbf{ else } \{\textbf{assume } \neg b\}) \\ =& sem_M(f_b(A), inl_M^n(\vec{U}, s) \ ; \ inl_M^n(\overrightarrow{U'}, w)) \cup f_{\neg b}(A) \end{aligned} \tag{31}$$

Using

$$\forall \varphi \in A. semB_M(\{\varphi\} \oplus deps(n+1), \textbf{exhale } I \ ; \textbf{ havoc } \overrightarrow{l}) = \{\overline{h}_{\overrightarrow{l}}(|\varphi|)\} \oplus h(\overrightarrow{l}) \oplus deps(n+1)$$

we obtain

$$\begin{aligned} & semB_M(A \oplus deps(n+1), w) \\ =& sem_M(A \oplus deps(n+1), \textbf{exhale } I \ ; \textbf{ havoc } \overrightarrow{l} \ ; \textbf{ assign } 0 \leq dep \leq n+1 \ ; \textbf{ inhale } I \ ; \textbf{ assume } \neg b \ ; \ dep \leftarrow n+1) \\ =& \bigcup_{\varphi \in A} sem_M(\{\overline{h}_{\overrightarrow{l}}(|\varphi|)\} \oplus h(\overrightarrow{l}) \oplus deps(n+1), \textbf{assign } dep = n+1 \ ; \textbf{ inhale } I \ ; \textbf{ assume } \neg b \ ; \ dep \leftarrow n+1) \qquad (S_1) \\ \cup & \bigcup_{\varphi \in A} sem_M(\{\overline{h}_{\overrightarrow{l}}(|\varphi|)\} \oplus h(\overrightarrow{l}) \oplus deps(n+1), \textbf{assign } 0 \leq dep \leq n \ ; \textbf{ inhale } I \ ; \textbf{ assume } \neg b \ ; \ dep \leftarrow n+1) \qquad (S_2) \end{aligned}$$

We will prove the following points:

- **Point 3.1**: $\pi_{\vec{V}}(S_1) = \pi_{\vec{V}}(f_{\neg b}(A))$
- **Point 3.2**: $\pi_{\vec{V}}(S_2) = \pi_{\vec{V}}(sem_M(A', inl_M^n(\overrightarrow{U'}, w)))$

Using these two points, we get

$$\begin{aligned} \pi_{\vec{V}}(sem_M(A, w)) =& \pi_{\vec{V}}(S_1) \cup \pi_{\vec{V}}(S_2) \\ =& \pi_{\vec{V}}(f_{\neg b}(A)) \cup \pi_{\vec{V}}(sem_M(A', inl_M^n(\overrightarrow{U'}, w))) \\ =& \pi_{\vec{V}}(f_{\neg b}(A)) \cup \pi_{\vec{V}}(sem_M(f_b(A), inl_M^n(\vec{U}, s) \ ; \ inl_M^n(\overrightarrow{U'}, w))) \\ =& \pi_{\vec{V}}(sem_M(A, inl_M^{n+1}(\overrightarrow{U'}, w))) \end{aligned}$$

9.6 Proving points 2.1 and 3.1

Let $\varphi \in A$.

$$\begin{aligned} & sem_M(\{\overline{h}_{\overrightarrow{l}}(|\varphi|)\} \oplus h(\overrightarrow{l}) \oplus deps(n+1), \textbf{assign } dep = n+1 \ ; \textbf{ inhale } I) \\ =& \{\overline{h}_{\overrightarrow{l}}(|\varphi|)\} \oplus h(\overrightarrow{l}) \oplus deps(n+1) \oplus Inh(I) \\ =& \{\overline{h}_{\overrightarrow{l}}(|\varphi|)\} \oplus h(\overrightarrow{l}) \oplus deps(n+1) \oplus \pi_{\vec{V}}(A) \qquad \text{(Well-annotated hypothesis)} \end{aligned}$$

Then

$$\pi_{\vec{V}}(sem_M(\{\overline{h}_{\overrightarrow{l}}(|\varphi|)\} \oplus h(\overrightarrow{l}) \oplus deps(n+1), \textbf{assign } dep = n+1 \ ; \textbf{ inhale } I)) = \pi_{\vec{V}}(A)$$

We then have

$$\pi_{\vec{V}}(\{\varphi\}) \subseteq \pi_{\vec{V}}(sem_M(\{\overline{h}_{\overrightarrow{l}}(|\varphi|)\} \oplus h(\overrightarrow{l}) \oplus deps(n+1), \textbf{assign } dep = n+1 \ ; \textbf{ inhale } I)) \subseteq \pi_{\vec{V}}(A)$$

Thus, we get

$$\pi_{\vec{V}}(sem_M(A \oplus deps(n+1), \textbf{exhale } I \ ; \textbf{ havoc } \overrightarrow{l} \ ; \textbf{ assign } 0 \leq dep \leq n+1 \ ; \textbf{ inhale } I)) = \pi_{\vec{V}}(A)$$

Then, since $dep \notin \vec{V}$ and $read(\textbf{assume } \neg b) \subseteq \vec{V}$,

$$\pi_{\vec{V}}(S_1) = \pi_{\vec{V}}(f_{\neg b}(A))$$

which proves **Point 3.1**

To prove points 1.2, 2.2, 3.2, we need to distinguish two cases.

9.7 First case: $A' \neq \varnothing$

In this case, there exists $\varphi' \in A'$. Since loop bodies cannot create variables (hypothesis), $\sigma(\varphi) = \sigma(\varphi')$. Moreover, by definition of A' and of $\overrightarrow{l}$, we have that φ and φ' are equal outside of $\overrightarrow{l}$, namely $\overline{h}_{\overrightarrow{l}}(|\varphi|) = \overline{h}_{\overrightarrow{l}}(|\varphi'|)$

9.7.1 Proving point 1.2

We have $semB_M(\{|\varphi|\} \oplus deps(n+1), \textbf{havoc } \overrightarrow{l}) = semB_M(\{|\varphi'|\} \oplus deps(n+1), \textbf{havoc } \overrightarrow{l})$. We can conclude using $verB_M(A' \oplus deps(n+1), w)$ (Equation 28).

9.7.2 Proving point 2.2 and 3.2

Since an iteration cannot create new variables, we have

$$\{\overline{h}_{\overrightarrow{l}}(|\varphi|)\} \oplus h(\overrightarrow{l}) = \{\overline{h}_{\overrightarrow{l}}(|\varphi'|)\} \oplus h(\overrightarrow{l})$$

From Equation 28, we get

$$\forall \varphi \in A.ver_M(\{\varphi\}, \textbf{exhale } I \text{ ; } \textbf{havoc } \overrightarrow{l} \text{ ; } \textbf{assign } 0 \leq dep \leq n \text{ ; } \textbf{inhale } I \text{ ; } \textbf{assume } \neg b \text{ ; } dep \leftarrow n+1)$$

and from Equation 29, we get

$$\pi_{\overrightarrow{V}}(S_2) = \pi_{\overrightarrow{V}}(sem_M(A', inl^n_M(\overrightarrow{U'}, w)))$$

9.8 Second case: $A' = \varnothing$

By induction over *wellAnnot*, we get:

$$\forall i \leq n. \overline{h}_{\overrightarrow{V}}(|\varphi|) \oplus h(\overrightarrow{V}) \oplus deps(i) \oplus Inh(I) = \varnothing$$

Since $\overrightarrow{l} \subseteq modif(s) \subseteq read(s) \subseteq \overrightarrow{V}$, we have that $\overline{h}_{\overrightarrow{l}}(|\varphi|) \oplus h(\overrightarrow{l}) \subseteq \overline{h}_{\overrightarrow{V}}(|\varphi|) \oplus h(\overrightarrow{V})$, thus

$$\forall i \leq n. \overline{h}_{\overrightarrow{l}}(|\varphi|) \oplus h(\overrightarrow{l}) \oplus deps(i) \oplus Inh(I) = \varnothing$$

Points 1.2 and **2.2** follow immediately.

9.8.1 Proving point 3.2

:

$$\begin{aligned}
&\pi_{\overrightarrow{V}}(S_2) \\
=&\pi_{\overrightarrow{V}}(\bigcup_{\varphi \in A} sem_M(\{\overline{h}_{\overrightarrow{l}}(|\varphi|)\} \oplus h(\overrightarrow{l}) \oplus deps(n+1), \textbf{assign } 0 \leq dep \leq n \text{ ; } \textbf{inhale } I \text{ ; } \textbf{assume } \neg b \text{ ; } dep \leftarrow n+1)) \\
=&\pi_{\overrightarrow{V}}(\bigcup_{\varphi \in A} \varnothing) \\
=&\pi_{\overrightarrow{V}}(\varnothing) \\
=&\pi_{\overrightarrow{V}}(sem_M(\varnothing, inl^n_M(\overrightarrow{U'}, w))) \\
=&\pi_{\overrightarrow{V}}(sem_M(A', inl^n_M(\overrightarrow{U'}, w)))
\end{aligned}$$

□