A compositional trace semantics for Orc

Dimitrios Vardoulakis and Mitchell Wand

Northeastern University
dimvar@ccs.neu.edu wand@ccs.neu.edu

Abstract. Orc [9] is a language for task orchestration. It has a small set of primitives, but sufficient to express many useful programs succinctly. We identify an ambiguity in the trace semantics of Kitchin et al. [9]. We give possible interpretations of the ambiguous definition and show that the semantics is not adequate regardless of the interpretation. We remedy this situation by providing new operational and denotational semantics with a better treatment of variable binding, and proving an adequacy theorem to relate them. Also, we investigate strong bisimulation in Orc and show that bisimulation implies trace equivalence but not vice versa.

1 Introduction

Orc [9] is a concurrent programming language for web-service orchestration. It is small yet usefully programmable, making it a good vehicle for the study of distributed processes in the presence of timeouts and communication failures. Orc uses autonomous computing units called *sites* to perform sequential computation and other basic services. It then provides operators to coordinate the execution of sites and build larger processes.

The question of the practical applicability of Orc is outside the scope of this paper. Popular concurrent programming patterns like fork-join parallelism can be coded in Orc, and also the workflow patterns of van der Aalst et al. [12]. The practical aspects of the language are discussed in [6, 9, 11]. Here, we will discuss the formal properties of Orc.

- The existing trace semantics for Orc [9] is ambiguous when there is a naming conflict between free and bound variables. We resolve the ambiguity and show that the semantics is not adequate.
- We suggest that dynamic binding of variables be prohibited because it invalidates an equivalence between Orc processes proved in [9].
- We provide new operational and denotational semantics which fix the aforementioned problems and prove an adequacy theorem to relate them.
- We investigate strong bisimulation in Orc and show that it is a congruence. We use it to prove useful equivalences between Orc processes. Last, we show that strong bisimulation implies trace equivalence but not vice versa.

This paper is organized as follows. We give a quick overview of Orc in the next section. Then we present the existing semantics [9] and its deficiencies in section 3. In section 4, we give our semantics for Orc. We study strong bisimulation in Orc in section 5. We discuss related work in section 6 and conclude in section 7.

2 Overview of Orc

The simplest Orc program is a *site call*. For example, the site call $\mathit{IsPrime}(N)$ sends the number N to a site named $\mathit{IsPrime}$. We imagine that this site will return *true* if N is prime and *false* otherwise. Similarly, we imagine that the result of the site call $\mathit{RedditFeed}(\mathit{today})$ will be a page of today's technical news. In Orc terminology, we use the word *publication* to refer to the result of a site call. A site may respond to a call at most once and it can also ignore the request. Note that the same site call at different times may publish different values.

In *symmetric composition* $(f \mid g)$ the two processes are evaluated in parallel and there is no interaction between them. The composite process publishes all the values published by f and g. For instance, the process $(\mathit{IsPrime}(N) \mid \mathit{RedditFeed}(\mathit{today}))$ can publish at most two values.

The *sequencing* operator $(f > x > g)$ is used to spawn threads. Process f starts running, and whenever f publishes some value v, an instance of g with v bound to x is launched in parallel. For example, $((\mathit{IsPrime}(N) \mid \mathit{RedditFeed}(\mathit{today})) > x > \mathit{Print}(x))$ may print twice, if both $\mathit{IsPrime}(N)$ and $\mathit{RedditFeed}(\mathit{today})$ publish. If f does not publish, g is not run.

Last, we can use the **where** operator to terminate a process after it publishes. The expression $(f \textbf{ where } x :\in g)$ starts evaluating f and g in parallel. However, the parts of f that depend on x block until x acquires a value. If g publishes, the value published is bound to x in f and g is terminated. Therefore, the expression $(\mathit{Print}(x) \textbf{ where } x :\in (\mathit{IsPrime}(N) \mid \mathit{RedditFeed}(\mathit{today})))$ will either print a boolean or today's technical news, maybe none, but not both.

The operators we saw up to now do not allow us to write recursive processes. To do that, we can define expressions like the following:

$$\mathit{DOS}(x) \triangleq \mathit{Ping}(x) \mid \mathit{DOS}(x)$$

This is a simple denial-of-service attack; the process $\mathit{DOS}(\mathit{ip})$ pings ip an unbounded number of times.

At this point we have explained the features of Orc informally and we can proceed to discuss its formal syntax and semantics.

3 The existing semantics of Orc and its deficiencies

3.1 Syntax - Operational Semantics

The syntax of Orc is shown in Fig. 1. An Orc program consists of a finite set of mutually recursive declarations and an expression that is evaluated with these declarations in scope. We use Δ to refer to the set of declarations. The terms "expression" and "process" will be used interchangeably.

The process $\mathbf{0}$ is the inert process. The actual parameter of a site call or a call to a defined expression is either a variable or a value. Values do not have types; they all belong to some generic set Val. Orc is not higher-order: a process is not a value. In what follows, we assume that processes are *well-formed*, i.e. do not contain $E_i(p)$ when there are fewer than i declarations in the program.

Program	$P ::= D_1, \ldots, D_k$ **in** e
Expression	$e ::= \mathbf{0} \mid M(p) \mid let(p) \mid E_i(p) \mid (e_1 \mid e_2) \mid e_1 > x > e_2 \mid e_1$ **where** $x :\in e_2$
Parameter	$p ::= x \mid v$
Declaration	$D_i ::= E_i(x) \triangleq e$

Fig. 1. Syntax of Orc

(SITECALL) $\dfrac{k \text{ fresh}}{M(v) \overset{M_k(v)}{\rightarrow} ?k}$

(SITERET) $?k \overset{k?v}{\rightarrow} let(v)$

(LET) $let(v) \overset{!v}{\rightarrow} \mathbf{0}$

(DEF) $\dfrac{(E_i(x) \triangleq f_i) \in \Delta}{E_i(p) \overset{\tau}{\rightarrow} [p/x]f_i}$

(SYM1) $\dfrac{f \overset{a}{\rightarrow} f'}{f \mid g \overset{a}{\rightarrow} f' \mid g}$

(SYM2) $\dfrac{g \overset{a}{\rightarrow} g'}{f \mid g \overset{a}{\rightarrow} f \mid g'}$

(SEQ1N) $\dfrac{f \overset{a}{\rightarrow} f' \qquad a \neq !v}{f > x > g \overset{a}{\rightarrow} f' > x > g}$

(SEQ1V) $\dfrac{f \overset{!v}{\rightarrow} f'}{f > x > g \overset{\tau}{\rightarrow} (f' > x > g) \mid [v/x]g}$

(ASYM1N) $\dfrac{f \overset{a}{\rightarrow} f'}{f \textbf{ where } x :\in g \overset{a}{\rightarrow} f' \textbf{ where } x :\in g}$

(ASYM1V) $\dfrac{g \overset{!v}{\rightarrow} g'}{f \textbf{ where } x :\in g \overset{\tau}{\rightarrow} [v/x]f}$

(ASYM2) $\dfrac{g \overset{a}{\rightarrow} g' \qquad a \neq !v}{f \textbf{ where } x :\in g \overset{a}{\rightarrow} f \textbf{ where } x :\in g'}$

Fig. 2. Existing operational semantics for Orc [9]

The operational semantics uses labeled transitions (Fig. 2). The metavariables f, g range over processes. Every transition is of the form $f \overset{a}{\rightarrow} f'$, meaning that process f takes a step to f' with event a. The events that occur during transitions are publications, internal events, site calls and site responses:

$$BaseEvent ::= !v \mid \tau \mid M_k(v) \mid k?v$$

Let's take a closer look at the rules. When process $M(v)$ calls site M with value v, a site call event occurs and a fresh handle k is allocated to identify the call (rule SITECALL). The resulting process $?k$ is just an idle thread waiting for an answer to the call with handle k. It is a necessary addition to the syntax to represent intermediate state.

If the site replies with some value w, $?k$ performs a site response event $k?w$ and becomes $let(w)$, as shown in rule SITERET. By rule LET, $let(w)$ publishes w and becomes $\mathbf{0}$, which has no further transitions.

None of the above steps is guaranteed to happen; $M(v)$ may delay the site call to M indefinitely, if the call happens M may never respond, and if it responds the value may not be published.

Defined expressions $E_i(p)$ are called by name (rule DEF). The actual parameter p is substituted for x in the body of E_i and the process continues as $[p/x]f_i$. This substitution is marked by an internal event τ.

$$
\begin{array}{ll}
(let(y) \mid let(2)) >x> M(x) \xrightarrow{\tau} & \text{by LET, SEQ1V} \\
((let(y) \mid \mathbf{0}) >x> M(x)) \mid M(2) \xrightarrow{M_k(2)} & \text{by SITECALL, SYM2} \\
((let(y) \mid \mathbf{0}) >x> M(x)) \mid ?k \xrightarrow{k?11} & \text{by SITERET, SYM2} \\
((let(y) \mid \mathbf{0}) >x> M(x)) \mid let(11) \xrightarrow{!11} & \text{by LET, SYM2} \\
((let(y) \mid \mathbf{0}) >x> M(x)) \mid \mathbf{0} &
\end{array}
$$

Fig. 3. Possible evaluation of $(let(y) \mid let(2)) >x> M(x)$

The rules for symmetric composition are simple; $f \mid g$ takes a step if either f or g takes a step. The steps of the sub-processes can be interleaved arbitrarily.

Process $f >x> g$ takes a step if f takes a step (rule SEQ1N). If f publishes v the process performs an internal event and launches a new instance of g in parallel (rule SEQ1V). We can think of x as an implicit communication channel between f and g.[1]

In asymmetric composition f **where** $x :\in g$, f and g execute in parallel unless g publishes. Then, g is terminated and the published value v is communicated via x to f (rule ASYM1V). Rule ASYM2 shows the non-publication steps of g, and ASYM1N shows the steps of f. Note that a $let(x)$ or a site call $M(x)$ in f will block waiting for a publication from g.

The example in Fig. 3 illustrates the use of some of the rules. Observe that processes can evaluate even when they have free variables.

Using the rules of Fig. 2, $M(x)$ has no transitions. It behaves like $\mathbf{0}$. However, in a context that can provide a value for x (see Fig. 3) $M(x)$ can publish and $\mathbf{0}$ cannot. To model this behavior, Kitchin et al. add one more rule:

$$\text{(SUBST)} \quad f \xrightarrow{[v/x]} [v/x]f$$

We call this new event a *receive* event.[2] Any process f can perform any receive step, even for variables not free in f (of course then $[v/x]f = f$). The constraint is that the SUBST rule cannot be applied to parts of an expression, in other words the event 'a' in the previous rules cannot be a receive event for any variable.

The reflexive and transitive closure of the transition relation is called *execution*:

Definition 1 (Execution). *t is an execution of f i.e. $f \xrightarrow{t}{}^* f'$ iff*
- *$t = \varepsilon$ and $f \equiv f'$, or*
- *$t = a\,t'$ and for some f'', $f \xrightarrow{a} f''$ and $f'' \xrightarrow{t'}{}^* f'$*

For instance, some executions of $let(x)$ are: $[2/x]\,[1/x]\,!2, \quad [3/y]\,[2/x]\,!2$

If t is a sequence of events then $t\backslash a$ is the sequence of events obtained from t when all instances of event a are removed.

Definition 2. *The trace set $\langle f \rangle$ of a process f is* $\{\, t\backslash\tau \mid t \textit{ is an execution of } f\}$

For example, every trace of $M(v)$ is a prefix of $\sigma_1\, M_k(v)\, \sigma_2\, k?w\, \sigma_3\, !w\, \sigma_4$ where $\sigma_1, \ldots, \sigma_4$ are arbitrary sequences of receives and w is an arbitrary value.

[1] versus the explicit prefix form $x(y).P$ of the π-calculus.

[2] This was called *substitution* event in [9]

3.2 Trace Semantics

Kitchin et al. attempt to provide a denotational semantics for processes by overloading the Orc combinators to work on trace sets. They define $T_1 \mid T_2$, $T_1 > x > T_2$, and T_1 **where** $x :\in T_2$ as follows.

Symmetric Composition

Definition 3 (Merge). *For traces t_1 and t_2, $t_1 \mid t_2$ is the set of all t such that*

- *t_1 and t_2 are subsequences of t and every event of t belongs to at least one of t_1 and t_2*
- *every common event of t (i.e. an event that belongs to both t_1 and t_2) is a receive event*
- *if t_1 and t_2 contain receives for the same variable x, the first receive for x in both t_1 and t_2 is a common event of t*

For example, if $t_1 = [1/x]\,!1$, $t_2 = [1/x]\,[4/x]\,M_k(4)$, $t_3 = [2/x]\,[11/y]$ then $(t_1 \mid t_2)$ contains three elements, including $[1/x]\,[4/x]\,!1\,M_k(4)$, and $(t_2 \mid t_3)$ is empty.

For trace sets, define $T_1 \mid T_2 = \bigcup_{t_1 \in T_1,\, t_2 \in T_2} t_1 \mid t_2$.

Sequencing

Define the operator: $T \upharpoonright [v/x] = \{\, t \mid [v/x]\, t \in T\}$. This selects the traces in T that start with $[v/x]$ and removes the leading receive event from these traces. For sequences of receives, define inductively:

$$T \upharpoonright \varepsilon = T$$
$$T \upharpoonright ([v/x]\,\sigma) = (T \upharpoonright [v/x]) \upharpoonright \sigma$$

Also, when a trace t has no publications we write $\bar{P}(t)$ and when t has no receives for x we write $\bar{R}(x,t)$.

Definition 4. *For trace s and trace set T, define the set $s > x > T$ by:*

$$\begin{cases} \{s\} & \bar{P}(s) \\ s_1((s_2 > x > T') \mid (T' \upharpoonright [u/x])) & s = s_1 !u\, s_2,\ \bar{P}(s_1), \\ & D \text{ is the sequence of receives in } s_1,\ T' = T \upharpoonright D \end{cases}$$

Note: Any receive event $[v/x]$ in s is unrelated to x in $(s > x > T)$

For trace sets, define $T_1 > x > T_2 = \bigcup_{s \in T_1} s > x > T_2$.

Every trace s of f that does not publish is also a trace of $\langle f \rangle > x > \langle g \rangle$. Moreover, if s contains a publication, an instance of g is launched in parallel and the remaining transitions of f may spawn more instances of g. For example, consider $\langle let(y) \rangle > x > \langle let(y) \mid let(x) \rangle$. The trace $([2/y]\,!2)$ is in $\langle let(y) \rangle$ and D is $[2/y]$. Also, $([2/y]\,[2/x]\,!2\,!2) \in \langle let(y) \mid let(x) \rangle$. Therefore, $([2/x]\,!2\,!2) \in \langle let(y) \mid let(x) \rangle \upharpoonright D$ which gives $(!2\,!2) \in T' \upharpoonright [2/x]$. Hence, $([2/y]\,!2\,!2) \in \langle let(y) \rangle > x > \langle let(y) \mid let(x) \rangle$.

The note in the definition of $s > x > T$ which we copy directly from [9] is ambiguous; what happens if s contains an event $[v/x]$? We discuss possible interpretations of the note in the following section.

Asymmetric Composition

Definition 5. *For traces t_1 and t_2, define the set t_1 **where** $x :\in t_2$ by:*

$$\begin{cases} t_1 \mid t_2 & \bar{P}(t_2) \\ (t_{11} \mid t_{21})t_{12} & t_1 \equiv t_{11}[v/x]t_{12},\ \bar{R}(x, t_{11}) \\ & t_2 \equiv t_{21}!v\, t_{22},\ \bar{P}(t_{21}) \\ \emptyset & \text{otherwise} \end{cases}$$

*Note: Any receive event $[v/x]$ in t_2 is unrelated to x in (t_1 **where** $x :\in t_2$)*

For trace sets, define $\langle f\rangle$ **where** $x :\in \langle g\rangle = \bigcup_{t_1\in\langle f\rangle,\, t_2\in\langle g\rangle} t_1$ **where** $x :\in t_2$.

If t_2 does not publish, asymmetric composition is like symmetric composition. If it publishes v and t_1 receives v, the part of t_2 prior to the publication is merged with the part of t_1 prior to the receive; followed by the rest of t_1. The rest of t_2 is discarded. The third branch disallows the creation of nonsensical traces that combine a t_1 that receives v_1 for x with a t_2 that publishes v_2.

Like sequencing, the definition of t_1 **where** $x :\in t_2$ is ambiguous about the treatment of receives for x in t_2.

3.3 Problems of Compositionality

To show that these definitions give a compositional semantics, Kitchin et al. make the following claims:

Claim. 1. $\langle f \mid g\rangle = \langle f\rangle \mid \langle g\rangle$
2. $\langle f > x > g\rangle = \langle f\rangle > x > \langle g\rangle$
3. $\langle f$ **where** $x :\in g\rangle = \langle f\rangle$ **where** $x :\in \langle g\rangle$

We believe Claim 1 is true, but Claims 2 and 3 are problematic.

Sequencing

The truth of Claim 2 depends on the interpretation of the ambiguous note.

1. Rename the bound variable x to avoid naming conflicts:
   Let $h = let(1) > x > \mathbf{0}$. The trace $[3/x]$ is in $\langle let(1)\rangle$. Therefore, we pick a fresh variable y and alpha-rename every event $[v/x]$ in $\langle \mathbf{0}\rangle$ to $[v/y]$. Let Z be the set we obtain after the alpha-renaming. Then, the set $[3/x] > x > \langle \mathbf{0}\rangle$ is defined to be equal to $[3/x] > y > Z$. By rule SUBST however, $\langle \mathbf{0}\rangle$ contains every finite sequence of receives, so there is no fresh variable to pick for the alpha-renaming; by this interpretation the set $[3/x] > x > \langle \mathbf{0}\rangle$ is undefined.
2. Receive events for x in s are not allowed in $s > x > T$:
   By this interpretation, the definition of $s > x > T$ becomes

$$\begin{cases} \{s\} & \bar{P}(s), \bar{R}(x, s) \\ s_1((s_2 > x > T') \mid (T' \restriction [u/x])) & s = s_1!u\, s_2,\ \bar{R}(x, s),\ \bar{P}(s_1),\ D \text{ is the} \\ & \text{sequence of receives in } s_1,\ T' = T \restriction D \\ \emptyset & \text{otherwise} \end{cases}$$

   Let $h = M(1) > x > let(x)$. By rules SUBST, SITECALL and SEQ1N,

$[3/x]\, M_k(1) \in \langle h \rangle$. Let $t = [3/x]\, M_k(1)$. We prove by contradiction that $t \notin (\langle M(1) \rangle > x > \langle let(x) \rangle)$, hence $\langle f > x > g \rangle \neq \langle f \rangle > x > \langle g \rangle$. Assume that $t \in (\langle M(1) \rangle > x > \langle let(x) \rangle)$. Then, there exists $s \in \langle M(1) \rangle$ such that $t \in (s > x > \langle let(x) \rangle)$.

a) If the first branch of the definition was used to produce t then $t = s$ which gives $\bar{R}(x, t)$, a contradiction.

b) If the second branch of the definition was used, then s is of the form $(\sigma_1\, M_k(1)\, \sigma_2\, k?w\, \sigma_3\, !w\, \sigma_4)$ where $\sigma_1, \ldots, \sigma_4$ are arbitrary sequences of receive events for variables different from x. But then, σ_1 must be $[3/x]$ which is a contradiction because $\bar{R}(x, s)$. We conclude that there is no $s \in \langle M(1) \rangle$ such that $t \in (s > x > \langle let(x) \rangle)$.

3. The note is simply a reminder that receives for x in s and receives for x in the traces of T refer to different variables, and has no other impact:
In this interpretation, the definition of $s > x > T$ is not influenced by the note; receives for x in s are treated like receives for other variables. Let $h = let(2) > x > let(x)$, $s = [1/x]\, !2$, $t = [1/x]\, [2/x]\, !1$. Clearly, $s \in \langle let(2) \rangle$ and the sequence of receives in s is $[1/x]$.
Also, $t \in \langle let(x) \rangle$ and $\{t\} \restriction [1/x] = \{[2/x]\, !1\} \Rightarrow ([2/x]\, !1) \in T' \Rightarrow \{[2/x]\, !1\} \restriction [2/x] = \{!1\}$. Then, $([1/x]\, !1) \in \langle let(2) \rangle > x > \langle let(x) \rangle$. But this trace cannot be produced by the operational semantics of h; every operational trace of h is of the form $(\sigma_1\, !2\, \sigma_2)$ where σ_1 and σ_2 are arbitrary sequences of receives. Thus, $\langle f > x > g \rangle \neq \langle f \rangle > x > \langle g \rangle$

Asymmetric Composition

Claim 3 is false independent of the note, as the following simple counterexample shows. Let $h = let(x)$ **where** $x :\in \mathbf{0}$. The only operational rule that applies to h is SUBST, which takes h to itself. This means that a trace of h can consist only of receive events. By SUBST and LET, $t = ([2/x]\, !2) \in \langle let(x) \rangle$ and also $\varepsilon \in \langle \mathbf{0} \rangle$. Then, $(([2/x]\, !2)$ **where** $x :\in \varepsilon) = (([2/x]\, !2) \mid \varepsilon) = \{[2/x]\, !2\}$ which yields $([2/x]\, !2) \in (\langle let(x) \rangle$ **where** $x :\in \langle \mathbf{0} \rangle)$. Clearly, $t \notin \langle h \rangle$. Therefore, $\langle f$ **where** $x :\in g \rangle \neq \langle f \rangle$ **where** $x :\in \langle g \rangle$

Dynamic Binding

Consider the defined expression $E(x) \triangleq e$. Kitchin et al. [9] do not impose any constraint on e, so it may contain variables other than x. In this case, dynamic binding can take place during the execution of a process. This invalidates a bisimulation result in [9], namely that when $x \notin \mathrm{fv}(g)$

$$(f \mid g) \textbf{ where } x :\in h \;\sim\; (f \textbf{ where } x :\in h) \mid g$$

Let $E(x) \triangleq let(y)$, $f_1 = (\mathbf{0} \mid E(2))$ **where** $y :\in let(1)$, $f_2 = (\mathbf{0}$ **where** $y :\in let(1)) \mid E(2)$. Then $\tau\, \tau\, !1$ is an execution of f_1 but not of f_2 because in any execution of f_2 a receive for y must precede the publication. The details of this are left to the reader.

(SITEC) $$\frac{k \text{ fresh}}{\Delta, \Gamma \vdash\ M(v) \overset{M_k(v)}{\rightarrow} ?k}$$

(SITECV) $$\frac{\Gamma(x) = v}{\Delta, \Gamma \vdash\ M(x) \overset{[v/x]}{\rightarrow} M(v)}$$

(SITER) $$\frac{}{\Delta, \Gamma \vdash ?k \overset{k?v}{\rightarrow} let(v)}$$

(LET) $$\frac{}{\Delta, \Gamma \vdash\ let(v) \overset{!v}{\rightarrow} \mathbf{0}}$$

(LETV) $$\frac{\Gamma(x) = v}{\Delta, \Gamma \vdash\ let(x) \overset{[v/x]}{\rightarrow} let(v)}$$

(SYM-L) $$\frac{\Delta, \Gamma \vdash\ f \overset{a}{\rightarrow} f'}{\Delta, \Gamma \vdash\ f \mid g \overset{a}{\rightarrow} f' \mid g}$$

(SYM-R) $$\frac{\Delta, \Gamma \vdash\ g \overset{a}{\rightarrow} g'}{\Delta, \Gamma \vdash\ f \mid g \overset{a}{\rightarrow} f \mid g'}$$

(DEF) $$\frac{(E_i(x) \triangleq f_i) \in \Delta}{\Delta, \Gamma \vdash\ E_i(v) \overset{\tau}{\rightarrow} [v/x]f_i}$$

(DEFV) $$\frac{(E_i(x) \triangleq f_i) \in \Delta \qquad \Gamma(x) = v}{\Delta, \Gamma \vdash\ E_i(x) \overset{[v/x]}{\rightarrow} E_i(v)}$$

(SEQ) $$\frac{\Delta, \Gamma \vdash\ f \overset{a}{\rightarrow} f' \qquad \bar{P}(a)}{\Delta, \Gamma \vdash\ f >x> g \overset{a}{\rightarrow} f' >x> g}$$

(SEQ-P) $$\frac{\Delta, \Gamma \vdash\ f \overset{!v}{\rightarrow} f'}{\Delta, \Gamma \vdash\ f >x> g \overset{\tau}{\rightarrow} (f' >x> g) \mid [v/x]g}$$

(ASYM-L) $$\frac{\Delta, \Gamma \vdash\ f \overset{a}{\rightarrow} f' \qquad \bar{R}(x, a)}{\Delta, \Gamma \vdash\ f \textbf{ where } x :\in g \overset{a}{\rightarrow} f' \textbf{ where } x :\in g}$$

(ASYM-R) $$\frac{\Delta, \Gamma \vdash\ g \overset{a}{\rightarrow} g' \qquad \bar{P}(a)}{\Delta, \Gamma \vdash\ f \textbf{ where } x :\in g \overset{a}{\rightarrow} f \textbf{ where } x :\in g'}$$

(ASYM-P) $$\frac{\Delta, \Gamma \vdash\ g \overset{!v}{\rightarrow} g'}{\Delta, \Gamma \vdash\ f \textbf{ where } x :\in g \overset{\tau}{\rightarrow} [v/x]f}$$

Fig. 4. Our operational semantics for Orc

Note: After the completion of this work, we contacted the authors of [9], who suggested corrections to their definitions. In $s >x> T$, D is the sequence of receive events in s_1 for variables other than x. In t_1 **where** $x :\in t_2$, add the side-condition $\bar{R}(x, t_1)$ to the first branch; the notes are no longer needed. Our counterexamples do not apply to the changed definitions; however we did not try to verify the adequacy of the fixed semantics.

Note that our counterexamples use processes where free and bound variables have distinct names, but since any process can take any receive step the naming conflict cannot be avoided in the traces.

4 New operational and trace semantics for Orc

4.1 Operational Semantics

Our operational semantics for Orc is shown in Fig. 4. Here is a summary of the changes.

No dynamic binding. The syntax of the language is unchanged. However, in a declaration $E_i(x) \triangleq f_i$ we demand that $\text{fv}(f_i) \subseteq \{x\}$. Hence, no dynamic binding can take place during process evaluation. This approach is also taken by Wehrman et al. [15].

$$\begin{array}{lll} \Delta, \Gamma \vdash & let(x) \textbf{ where } x :\in (M(x) \mid let(x)) \overset{[\text{“hi”}/x]}{\longrightarrow} & \text{by LET-VAR, SYM-R, ASYM-R} \\ & let(x) \textbf{ where } x :\in (M(x) \mid let(\text{“hi”})) \overset{\tau}{\longrightarrow} & \text{by LET, SYM-R, ASYM-P} \\ & let(\text{“hi”}) & \end{array}$$

Fig. 5. Possible evaluation when $(x, \text{“hi”}) \in \Gamma$

Defined expressions are called by value. Since we do not know of any Orc program where call-by-name functionality is absolutely necessary, we made this change because it simplifies the technical treatment.

A process f can take a $[v/x]$ step only when x is free in f. By thinking of variables as channels, we say that f can receive only on a channel it knows i.e. when x is free in f.

When x is not free in f a receive $[v/x]$ would leave f unchanged, therefore such receives can be harmlessly forbidden. Consequently, closed processes do not take any receive steps throughout their execution.

The condition $x \in \text{fv}(f)$ is necessary but not sufficient for a receive step, for example the process $\mathbf{0} >y> let(x)$ is inert.

Addition of an environment Γ. Let f take a $[v/x]$ step to f'. This means that if f is plugged in a process-context that can provide v for x, f can receive v and behave like f' (as in Fig. 3 for $M(x)$).

We use environments to model process contexts. An environment is a partial function from variables to values. The metavariable Γ ranges over environments. With this formulation, $M(x)$ can go to $M(v)$ only when (x, v) is in Γ, and is inert otherwise. Note that, unlike traditional environments in operational semantics, Γ can be non-empty at the beginning of the evaluation of a process and it remains unchanged throughout the evaluation. This is because Γ keeps track of the free variables in a process, but local binding is handled by substitution (e.g. rule SEQ-P).

By using Γ instead of a SUBST-like rule which can be applied to whole processes only, we do not need to differentiate between receives and base events.

$$Event ::= BaseEvent \mid [v/x]$$

So, the event 'a' in our rules refers to any event, not just to a base event. Also, observe that in ASYM-L f cannot proceed with a receive for x. Its parts that depend on x are blocked waiting for a publication from g. See Fig. 5 for a sample evaluation using the new operational semantics.

4.2 Denotational Semantics

We now present our denotational semantics for Orc, which is based on complete partial orders. The meaning of a process is a set of traces in the presence of environments for the declarations *Fenv* and variables *Env*:

$$[\![f]\!] : [Fenv \rightarrow [Env \rightarrow P]]$$

$$
\begin{array}{l}
[\![\mathbf{0}]\!] = \lambda\varphi.\lambda\rho.\{\varepsilon\} \\
[\![let(v)]\!] = \lambda\varphi.\lambda\rho.\{!v\}_{\mathrm{p}} \\
[\![let(x)]\!] = \lambda\varphi.\lambda\rho.\mathbf{case}\ \rho(x)\ \mathbf{of}\ \text{Absent} \Rightarrow \{\varepsilon\} \\
\qquad\qquad\qquad\qquad\qquad\qquad\quad v \quad\ \ \Rightarrow \{[v/x]\ !v\}_{\mathrm{p}} \\
[\![M(v)]\!] = \lambda\varphi.\lambda\rho.\{\, M_k(v)\ k?w\ !w \mid k\ \text{fresh}\,,\, w \in \mathit{Val}\}_{\mathrm{p}} \\
[\![M(x)]\!] = \lambda\varphi.\lambda\rho.\mathbf{case}\ \rho(x)\ \mathbf{of}\ \text{Absent} \Rightarrow \{\varepsilon\} \\
\qquad\qquad\qquad\qquad\qquad\qquad\quad v \quad\ \ \Rightarrow \{\, [v/x]\ M_k(v)\ k?w\ !w \mid k\ \text{fresh}\,,\, w \in \mathit{Val}\}_{\mathrm{p}} \\
[\![?k]\!] = \lambda\varphi.\lambda\rho.\{\, k?w\ !w \mid w \in \mathit{Val}\}_{\mathrm{p}} \\
[\![E_i(v)]\!] = \lambda\varphi.\lambda\rho.\{\, \tau\ t \mid t \in \varphi_i(v)\}_{\mathrm{p}} \\
[\![E_i(x)]\!] = \lambda\varphi.\lambda\rho.\mathbf{case}\ \rho(x)\ \mathbf{of}\ \text{Absent} \Rightarrow \{\varepsilon\} \\
\qquad\qquad\qquad\qquad\qquad\qquad\quad v \quad\ \ \Rightarrow \{\, [v/x]\ \tau\ t \mid t \in \varphi_i(v)\}_{\mathrm{p}} \\
[\![h \mid g]\!] = \lambda\varphi.\lambda\rho.\ [\![h]\!]\varphi\rho \parallel [\![g]\!]\varphi\rho \\
[\![h > x > g]\!] = \lambda\varphi.\lambda\rho. \bigcup_{s \in [\![h]\!]\varphi\rho} s \gg \lambda v.([\![g]\!]\varphi\rho[x = v]) \backslash [v/x] \\
[\![h\ \mathbf{where}\ x :\in g]\!] = \lambda\varphi.\lambda\rho.\ (\bigcup_{v \in \mathit{Val}} [\![h]\!]\varphi\rho[x = v]) <_x [\![g]\!]\varphi\rho
\end{array}
$$

Fig. 6. Trace Semantics of Orc

A trace is a (possibly empty) sequence of events. Unlike the previous trace semantics, internal events appear in traces. Trace sets are prefix-closed and ordered by inclusion. They are also non-empty because the empty trace ε is a trace of any process. Last, we consider traces of finite length only; an infinite trace is represented by the set of all its finite prefixes.

$$
\begin{array}{l}
\mathit{Traces} = \mathit{Event}^*, \text{ a discrete CPO.} \\
P = \{\, S \mid S \subseteq \mathit{Traces} \ \wedge\ S \neq \emptyset \ \wedge\ S \text{ is prefix-closed}\} \\
\mathit{Val} = \text{the set of all values, a discrete CPO.} \\
\mathit{Var} = \text{the set of all variable names, a discrete CPO.} \\
\mathit{Env} = [\,\mathit{Var} \rightarrow (\,\mathit{Val} \cup \{\text{Absent}\})] \\
\mathit{NoRecv} = \{\, S \mid S \in P \ \wedge\ \forall t \in S, x \in \mathit{Var}.\ \bar{R}(x,t)\} \\
\mathit{Fenv} = ([\,\mathit{Val} \rightarrow \mathit{NoRecv}])^k
\end{array}
$$

Consider a declaration $(E_i(x) \triangleq f_i)$. Since only x can be free in f_i, the traces of $E_i(v)$ do not contain any receives. *NoRecv* is a CPO with bottom element $\{\varepsilon\}$ and *Fenv* inherits its order from *NoRecv* in the usual way. We do not need names to refer to the declared processes, we can index them by the order of declaration.

The definitions of the meaning functions can be found in Fig. 6. Juxtaposition of traces means concatenation. Various auxiliary operators are defined in Fig. 7. The operations $t\backslash a$, $t_1 \parallel t_2$, t_{p} and $(t_1 <_x t_2)$ are lifted to trace sets in the obvious way.

We can easily establish the following properties of the meaning functions:

Theorem 1 (Prefix Closure of Trace Sets). *For all* f, φ, ρ, $[\![f]\!]\varphi\rho \in P$

Theorem 2 (Continuity of Denotations). *For all* f, $[\![f]\!]$ *is continuous.*

Lemma 1 (Substitution). $[\![[v/x]f]\!]\varphi\rho = ([\![f]\!]\varphi\rho[x = v])\backslash[v/x]$

Remove event 'a' from a trace:
$$t\backslash a \triangleq \begin{cases} \varepsilon & t = \varepsilon \\ t'\backslash a & t = at' \\ a'\,(t'\backslash a) & t = a't' \text{ and } a \neq a' \end{cases}$$

Merge:
$$t_1 \parallel t_2 \triangleq \begin{cases} \{t_1\} & t_2 = \varepsilon \\ \{t_2\} & t_1 = \varepsilon \\ a(t_1' \parallel t_2) \cup b(t_1 \parallel t_2') & t_1 = at_1' \text{ and } t_2 = bt_2' \end{cases}$$

Prefix-closure:
$$t_{\mathrm{p}} \triangleq \begin{cases} \{\varepsilon\} & t = \varepsilon \\ \{\varepsilon, a\} \cup a\, t'_{\mathrm{p}} & t = at' \end{cases}$$

Sequencing combinator:
$$s \gg F = \begin{cases} \{s\} & \bar{P}(s) \\ s_1\, \tau\, ((s_2 \gg F) \parallel F(v)) & s \equiv s_1!vs_2\,,\, \bar{P}(s_1) \end{cases}$$

Asymmetric combinator:
$$t_1 <_x t_2 = \begin{cases} t_1 \parallel t_2 & \bar{R}(x, t_1)\,,\, \bar{P}(t_2) \\ t_1 \parallel t_{21}\tau & \bar{R}(x, t_1)\,,\, t_2 \equiv t_{21}!v\, t_{22}\,,\, \bar{P}(t_{21}) \\ (t_{11} \parallel t_{21}\tau)(t_{12}\backslash[v/x]) & t_1 \equiv t_{11}[v/x]t_{12}\,,\, \bar{R}(x, t_{11})\,, \\ & t_2 \equiv t_{21}!v\, t_{22}\,,\, \bar{P}(t_{21}) \\ \{\varepsilon\} & \text{otherwise} \end{cases}$$

Empty environment ρ_0:
$\rho_0(x) = \text{Absent}$ for all x

Fig. 7. Various Definitions

One might expect $[\![[v/x]f]\!]\varphi\rho$ to be equal to $[\![f]\!]\varphi\rho[x = v]$. However, since in the latter v is provided by the environment we have to remove $[v/x]$ from f's traces in order to equate it with $[v/x]f$.

The proofs of these and all subsequent theorems can be found in [13]. Finally, we apply the usual fixed-point technique [16] to give the denotation of a set of declarations Δ: we define an *Fenv* transformer $\hat{\Delta}$ by

$$\hat{\Delta} = \lambda\varphi.(\lambda v.([\![f_1]\!]\varphi\rho_0[x = v])\backslash[v/x] \times \cdots \times \lambda v.([\![f_k]\!]\varphi\rho_0[x = v])\backslash[v/x])$$

$\hat{\Delta}$ is continuous, so we define $[\![\Delta]\!]$ as its least fixed point

$$[\![\Delta]\!] = \mathrm{fix}(\hat{\Delta})$$

To prove the correctness of our semantics we need to show that the executions of a process match its traces.

Theorem 3 (Adequacy).
If $\rho = \rho_0[x_1 = v_1]\dots[x_m = v_m],\ \Gamma = \{(x_1, v_1), \dots, (x_m, v_m)\}$ *then*

$$t \in [\![f]\!][\![\Delta]\!]\rho \quad \textit{iff} \quad \exists f'.\ \Delta, \Gamma \vdash\ f \ \xrightarrow{t}{}^{*} f'$$

The theorem is proved by induction on the length of t. It relies on the following lemma, which is proved by structural induction on f.

Lemma 2. *If* $\rho = \rho_0[x_1 = v_1]\dots[x_m = v_m],\ \Gamma = \{(x_1, v_1), \dots, (x_m, v_m)\}$ *then*

$$a\, t \in [\![f]\!][\![\Delta]\!]\rho \quad \textit{iff} \quad \exists f'.\ \Delta, \Gamma \vdash\ f \ \xrightarrow{a}\ f' \textit{ and } t \in [\![f']\!][\![\Delta]\!]\rho$$

Let's look at an interesting property concerning the publications of a process f. When a sub-process of f publishes, the publication is either masked as a τ and sent to another sub-process (SEQ-P, ASYM-P), or it is observed by f's context. Observable publications do not trigger other events of f. The next lemma shows that there is no causality between a publication and the events that follow it in a trace.

Lemma 3. *If* $s_1\, !v\, s_2 \in [\![f]\!][\![\Delta]\!]\rho$ *then* $s_1(!v \parallel s_2) \subseteq [\![f]\!][\![\Delta]\!]\rho$

4.3 Semantics insensitive to internal events

Any Orc process can be a building block of a larger process, e.g. $\mathit{IsPrime}(N)$ in $(\mathit{Print}(x)$ **where** $x :\in (\mathit{IsPrime}(N) \mid \mathit{RedditFeed}(\mathit{today})))$. In such situations, the internal events of a process are not observable by its context, in the sense that they do not entail communication between the process and the rest of the system. Instead, τ events represent communication that takes place *within* the process. Therefore, we would like to have a semantics insensitive to internal events:

Definition 6. $\{\!|f|\!\} \triangleq \lambda\varphi.\lambda\rho.[\![f]\!]\varphi\rho\backslash\tau$

One could also define $\{\!|f|\!\}$ compositionally and independent of $[\![f]\!]$ and then prove definition 6 as a theorem.

Obviously, $[\![f]\!] = [\![g]\!]$ implies $\{\!|f|\!\} = \{\!|g|\!\}$. Therefore, this semantics is less discriminating than the semantics in section 4.2. We can now prove the following equivalence, which is false in our original trace semantics:

Lemma 4. *For all* $f, \rho \quad \{\!|f|\!\}\{\!|\Delta|\!\}\rho = \{\!|f >x> \mathit{let}(x)|\!\}\{\!|\Delta|\!\}\rho$

5 Strong Bisimulation Congruences

In [9], Kitchin et al. state some useful equivalences between processes using strong bisimulation [10]. However, some of these equivalences are invalid because of dynamic binding in the declarations. Also, they do not show bisimulation to be a congruence and do not investigate the relation between bisimulation and trace equivalence. For our semantics, we define a family of strong bisimulation relations indexed by Δ:

For any Δ such that f, g, h are well-formed,

1. $f \mid \mathbf{0} \sim_\Delta f$
2. $f \mid g \sim_\Delta g \mid f$
3. $f \mid (g \mid h) \sim_\Delta (f \mid g) \mid h$
4. $(f \mid g) >x> h \sim_\Delta (f >x> h) \mid (g >x> h)$
5. $f >x> (g >y> h) \sim_\Delta (f >x> g) >y> h \quad$ if $x \notin \text{fv}(h)$
6. $(f \mid g)$ **where** $x :\in h \sim_\Delta (f$ **where** $x :\in h) \mid g \quad$ if $x \notin \text{fv}(g)$
7. $(f >y> g)$ **where** $x :\in h \sim_\Delta (f$ **where** $x :\in h) >y> g \quad$ if $x \notin \text{fv}(g)$
8. $(f$ **where** $x :\in g)$ **where** $y :\in h \sim_\Delta (f$ **where** $y :\in h)$ **where** $x :\in g$
   if $y \notin \text{fv}(g)$ and $x \notin \text{fv}(h)$

Fig. 8. Strongly Bisimilar Processes

Definition 7 (Δ-bisimulation). *Let Δ be a set of declarations. Then, a binary relation $\mathfrak{R}$ on processes is a Δ-bisimulation iff*

1. *$\mathfrak{R}$ is symmetric*
2. *for any $(f, g) \in \mathfrak{R}$ and for any Γ if $\Delta, \Gamma \vdash f \xrightarrow{a} f'$ then $\exists g'. \Delta, \Gamma \vdash g \xrightarrow{a} g'$ and $(f', g') \in \mathfrak{R}$*

Definition 8 (Largest Strong Bisimulation). $\sim_\Delta \triangleq \bigcup\{\, \mathfrak{R} \mid \mathfrak{R}$ *is a Δ-bisim.*$\}$

For different declaration sets we get different bisimulations. For example,

$$E_1(v) \sim_{\Delta_1} (let(v) >x> M(x)) \quad \text{for} \quad \Delta_1 = \{E_1(x) \triangleq M(x)\}$$

but

$$E_1(v) \not\sim_{\Delta_2} (let(v) >x> M(x)) \quad \text{for} \quad \Delta_2 = \{E_1(x) \triangleq \mathbf{0}\}$$

We can prove the equivalences in [9] using our new operational semantics (see Fig 8). Naturally, symmetric composition is commutative and associative (equiv. 2, 3). Symmetric composition can be distributed over sequencing because symmetrically composed processes do not communicate with each other (equiv. 4). Equivalence 6 verifies our intuition that a (**where** x)-context does not influence a process g if x is not free in g.

Lemma 5. *For any Δ, $\sim_\Delta$ is a congruence relation*

The proof proceeds by induction on contexts. By lemma 5, the equivalences of Fig. 8 become congruences automatically. Congruence is important in a concurrent setting, because we can replace a process in a system with a congruent process without affecting the behavior of the system. The following example illustrates congruences 1, 2 and 6 when $x \notin \text{fv}(g)$

$$g \ \textbf{where}\ x :\in h \ \sim_\Delta (\mathbf{0} \mid g) \ \textbf{where}\ x :\in h \ \sim_\Delta (\mathbf{0}\ \textbf{where}\ x :\in h) \mid g$$

Definition 7 is universally quantified over Γ. This helps establish a connection between strong bisimulation and trace equivalence:

Theorem 4. *If $f \sim_\Delta g$ then for any ρ,* $\quad [\![f]\!][\![\Delta]\!]\rho = [\![g]\!][\![\Delta]\!]\rho$

As one might expect, trace equivalence does not imply bisimilarity:
Let $f = let(y)$ **where** $y :\in (let(1) >x> (let(2) \mid let(3)))$
and $g = (let(y)$ **where** $y :\in let(x))$ **where** $x :\in (let(2) \mid let(3))$.
For any Δ, ρ we get $[\![f]\!][\![\Delta]\!]\rho = [\![g]\!][\![\Delta]\!]\rho = \{\tau\,\tau\,!2,\ \tau\,\tau\,!3\}_{\mathrm{p}}$.
Let $\mathfrak{R}$ be a Δ-bisimulation and $(f, g) \in \mathfrak{R}$. Then, g must be able to match the steps of f.
$\Delta, \Gamma \vdash\ f \ \stackrel{\tau}{\to}\ let(y)$ **where** $y :\in ((\mathbf{0} >x> (let(2) \mid let(3))) \mid (let(2) \mid let(3))) \equiv f'$
The possible τ transitions of g are
$\Delta, \Gamma \vdash\ g \ \stackrel{\tau}{\to}\ let(y)$ **where** $y :\in let(2) \equiv g'$
$\Delta, \Gamma \vdash\ g \ \stackrel{\tau}{\to}\ let(y)$ **where** $y :\in let(3) \equiv g''$
It should be obvious that $(f', g') \notin \mathfrak{R}$ and $(f', g'') \notin \mathfrak{R}$ because g', g'' have lost one publishing option while f' maintains both. Formally, by the contrapositive of theorem 4 we get $f' \not\sim_\Delta g'$ and $f' \not\sim_\Delta g''$ because their trace sets differ. Assuming that $\mathfrak{R}$ exists leads to a contradiction, therefore $f \not\sim_\Delta g$.

We now discuss a limitation of our semantics. Let $f_1 = let(y) >x> let(x)$, $f_2 = let(y) >x> let(y)$, $\Gamma = \{(y, 42)\}$. These processes exhibit similar behaviors in Γ, they can receive 42 and publish it. However, they are not bisimilar. The reason is that the right-hand-side of f_1 will receive 42 from the left-hand-side, whereas the right-hand-side of f_2 will receive 42 from the context. We know that this difference is unimportant because the value published by both will always be the same, but we cannot equate such processes using our operational semantics. A possible solution would be to propagate the receives with rules like:

$$(\text{SYM-L}')\ \frac{\Delta \vdash\ f \ \stackrel{[v/x]}{\to}\ f'}{\Delta \vdash\ f \mid g \ \stackrel{[v/x]}{\to}\ f' \mid [v/x]g}$$

We have not verified the correctness of this semantics. We opted for the simpler semantics and as a trade-off lost the ability to equate a small class of Orc processes.

6 Related Work

Task orchestration is related to various industrial standards for business transactions (e.g. WSBPEL [1], WSCDL [8]). Academics have also looked at other aspects of business transactions, such as compensations (see [2–5]). A formal specification for a subset of WSBPEL has been proposed as well [14].

The semantics in [9] and this paper are asynchronous. Misra et al. [11] augment the operational semantics of [9] with a synchronous semantics. This is an operational semantics that gives priority to internal events, thus allowing the possibility for processes to synchronize on external interactions. However, they do not give a denotational semantics, nor do they state any theorems. Hoare et al. [7] present a tree-based denotational semantics for Orc, and sketch an operational semantics based on the same trees. They prove a number of interesting denotational equivalences, but do not state any theorem relating the operational and denotational semantics. Wehrman et al. [15] have developed a timed semantics for Orc, but in their semantics the observable events are quite different; except publications, all other events are internal.

7 Conclusions

In this paper we presented operational and denotational semantics for Orc, a language for task orchestration. We proved an adequacy theorem, showing that the operational transitions of a process coincide with its denotational traces. This is not the case in [9], as demonstrated in section 3. We also discussed strong bisimulation in Orc and showed it to be a congruence. Finally, we showed that in Orc strong bisimulation is more discriminating than trace equivalence, which is also the case in other process calculi like CCS and the π-calculus.

References

1. Alexandre Alves, Assaf Arkin, et al. Web Services Business Process Execution Language version 2.0. Technical report, April 2007. http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.pdf.
2. Roberto Bruni, Hernán C. Melgratti, and Ugo Montanari. Theoretical Foundations for Compensations in Flow Composition Languages. In Jens Palsberg and Martín Abadi, editors, *Proceedings of the 32nd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 209–220. ACM, 2005.
3. Michael J. Butler and Carla Ferreira. A Process Compensation Language. In Wolfgang Grieskamp, Thomas Santen, and Bill Stoddart, editors, *Integrated Formal Methods, Second International Conference*, volume 1945 of *Lecture Notes in Computer Science*, pages 61–76. Springer, 2000.
4. Michael J. Butler and Carla Ferreira. An Operational Semantics for StAC, a Language for Modelling Long-Running Business Transactions. In Rocco De Nicola, Gian Luigi Ferrari, and Greg Meredith, editors, *COORDINATION*, volume 2949 of *Lecture Notes in Computer Science*, pages 87–104. Springer, 2004.
5. Michael J. Butler, C. A. R. Hoare, and Carla Ferreira. A Trace Semantics for Long-Running Transactions. In Ali E. Abdallah, Cliff B. Jones, and Jeff W. Sanders, editors, *Communicating Sequential Processes: The First 25 Years*, volume 3525 of *Lecture Notes in Computer Science*, pages 133–150. Springer, 2004.
6. William R. Cook, Sourabh Patwardhan, and Jayadev Misra. Workflow Patterns in Orc. In Paolo Ciancarini and Herbert Wiklicky, editors, *COORDINATION*, volume 4038 of *Lecture Notes in Computer Science*, pages 82–96. Springer, 2006.
7. C. A. R. Hoare, Galen Menzel, and Jayadev Misra. A Tree Semantics for an Orchestration Language, August 2004. Lecture Notes for NATO summer school, Marktoberdorf.
8. Nickolas Kavantzas, David Burdett, et al. Web Services Choreography Description Language version 1.0. Technical report, November 2005. http://www.w3.org/TR/ws-cdl-10/.
9. David Kitchin, William R. Cook, and Jayadev Misra. A Language for Task Orchestration and its Semantic Properties. In Christel Baier and Holger Hermanns, editors, *CONCUR*, volume 4137 of *Lecture Notes in Computer Science*, pages 477–491. Springer, 2006.

10. Robin Milner. Operational and Algebraic Semantics of Concurrent Processes. In Jan van Leeuwen, editor, *Handbook of Theoretical Computer Science, Volume B: Formal Models and Semantics (B)*, pages 1201–1242. MIT Press/Elsevier, 1990.
11. Jayadev Misra and William R. Cook. Computation Orchestration: A Basis for Wide-Area Computing. *Software and Systems Modeling*, 6(1):83–110, 2007.
12. Wil M. P. van der Aalst, Arthur H. M. ter Hofstede, Bartek Kiepuszewski, and Alistair P. Barros. Workflow Patterns. *Distributed and Parallel Databases*, 14(1):5–51, 2003.
13. Dimitrios Vardoulakis and Mitchell Wand. A Compositional Trace Semantics for Orc. Technical report, Northeastern University, College of Computer and Information Science, March 2008. http://www.ccs.neu.edu/∼dimvar/papers/orc-coord.pdf.
14. Mirko Viroli. Towards a Formal Foundation to Orchestration Languages. *Electr. Notes Theor. Comput. Sci.*, 105:51–71, 2004.
15. Ian Wehrman, David Kitchin, William R. Cook, and Jayadev Misra. A Timed Semantics of Orc. unpublished.
16. Glynn Winskel. *The Formal Semantics of Programming Languages.* MIT Press, Cambridge, MA, 1993.

A Various Definitions

Definition 9. *Concatenate a trace and a trace-set*
$s\,T \triangleq \{\, s\,t \mid t \in T\}$

Definition 10. *Concatenate trace-sets*
$T_1\,T_2 \triangleq \{\, t_1 t_2 \mid t_1 \in T_1, t_2 \in T_2\}$

Definition 11. *Remove event 'a' from a trace*
$$t \backslash a \triangleq \begin{cases} \varepsilon & t = \varepsilon \\ t' \backslash a & t = at' \\ a'\,t' \backslash a & t = a't' \text{ and } a \neq a' \end{cases}$$

Definition 12. *Remove event from a trace-set*
$T \backslash a \triangleq \{\, t \backslash a \mid t \in T\}$

Definition 13. *Merge for traces*
$$t_1 \parallel t_2 \triangleq \begin{cases} \{t_1\} & t_2 = \varepsilon \\ \{t_2\} & t_1 = \varepsilon \\ a(t_1' \parallel t_2) \cup b(t_1 \parallel t_2') & t_1 = at_1' \text{ and } t_2 = bt_2' \end{cases}$$

Definition 14. *Merge for trace-sets*
$T_1 \parallel T_2 \triangleq \bigcup_{t_1 \in T_1, t_2 \in T_2} t_1 \parallel t_2$

Definition 15. *Prefixing*
$$t_{\mathrm{p}} \triangleq \begin{cases} \{\varepsilon\} & t = \varepsilon \\ \{\varepsilon, a\} \cup a\,t'_{\mathrm{p}} & t = at' \end{cases}$$

Definition 16. *Prefixing for trace-sets*
$S_{\mathrm{p}} \triangleq \bigcup_{s \in S} s_{\mathrm{p}}$

Definition 17. *Extend-env: Env $\times$ (Var $\times$ Val) $\rightarrow$ Env*
$\rho[x = u] \triangleq (\rho - \{(x, w)\}) \cup \{(x, u)\}$ *,where* $\rho(x) = w$

Definition 18. *Alternate merge*
$$t_1 \breve{\parallel} t_2 \triangleq \begin{cases} \{t_1\} & t_2 = \varepsilon \\ \{t_2\} & t_1 = \varepsilon \\ (t_1' \breve{\parallel} t_2)a \cup (t_1 \breve{\parallel} t_2')b & t_1 = t_1'a \text{ and } t_2 = t_2'b \end{cases}$$

Definition 19. *Alternate merge for trace-sets*
$T_1 \breve{\parallel} T_2 \triangleq \bigcup_{t_1 \in T_1, t_2 \in T_2} t_1 \breve{\parallel} t_2$

Note 6 *$\bar{P}(t)$ means that trace t contains no publications. $\bar{R}(x, t)$ means that trace t contains no receive events for x.*

Sequencing combinator:

$$s \gg F = \begin{cases} \{s\} & \bar{P}(s) \\ s_1\,\tau\,((s_2 \gg F) \parallel F(v)) & s \equiv s_1!vs_2\,,\bar{P}(s_1) \end{cases}$$

Asymmetric combinator:

$$t_1 <_x t_2 = \begin{cases} t_1 \parallel t_2 & \bar{R}(x,t_1)\,,\bar{P}(t_2) \\ t_1 \parallel t_{21}\tau & \bar{R}(x,t_1)\,,t_2 \equiv t_{21}!v\,t_{22}\,,\bar{P}(t_{21}) \\ (t_{11} \parallel t_{21}\tau)(t_{12}\backslash[v/x]) & t_1 \equiv t_{11}[v/x]t_{12}\,,\bar{R}(x,t_{11})\,, \\ & t_2 \equiv t_{21}!v\,t_{22}\,,\bar{P}(t_{21}) \\ \emptyset & \text{otherwise} \end{cases}$$

Asymmetric combinator for trace-sets:
$T_1 <_x T_2 = \bigcup_{t_1 \in T_1, t_2 \in T_2} t_1 <_x t_2$

Definition 20. $\rho_{-x}(y) = \begin{cases} \text{Absent} & y = x \\ \rho(y) & y \neq x \end{cases}$

Note 7 ρ_0 *is an environment such that* $\forall x.\rho_0(x) = \text{Absent}$

Note 8 $a \hat{\in} t$ *means that trace* t *contains event* a*.* $a \hat{\notin} t$ *means that trace* t *does not contain event* a*.*

Definition 21. *Ordering of pairs of integers*
$(i,j) \sqsubset (k,l)$ *when* $(i < k) \vee (i = k \wedge j < l)$

B Continuity Proofs

Lemma 9. *The union of prefix-closed sets is prefix-closed*

Lemma 10. *P is a CPO under inclusion*

Proof. Let $X \subseteq P$ be directed and $B = \bigcup_{S \in X} S$. Then, B is prefix-closed by Lemma 9 and is an ub of X. Let B' be an ub of X
$\implies \forall S \in X. S \subseteq B'$
$\implies \bigcup_{S \in X} S \subseteq B'$
$\implies \bigsqcup X = B$ □

Lemma 11. *Merge : $Pow(Traces) \times Pow(Traces) \rightarrow Pow(Traces)$ is continuous*

Proof. It suffices to show that it is continuous in each argument separately. Let $X \subseteq Pow(Traces)$ be directed, $T \in Pow(Traces)$
$(\bigsqcup X) \parallel T = (\bigcup_{S \in X} S) \parallel T$
$\triangleq \bigcup_{s \in (\bigcup_{S \in X} S)} \bigcup_{t \in T} s \parallel t$
$= \bigcup_{S \in X} \bigcup_{s \in S} \bigcup_{t \in T} s \parallel t$
$\triangleq \bigcup_{S \in X} (S \parallel T)$
$= \bigsqcup_{S \in X} (S \parallel T)$
The proof is similar for the right argument □

Lemma 12. *Extend-env is continuous*

Note 13 *$[Val \rightarrow NoRecv]$ is a CPO and if $X \subseteq [Val \rightarrow NoRecv]$ is directed, then $\bigsqcup X = \lambda v. \bigsqcup_{f \in X} f(v) = \lambda v. \bigcup_{f \in X} f(v)$*

Note 14 *Fenv is a CPO and if $X \subseteq Fenv$ is directed, then*
$\bigsqcup X = (\lambda v. \bigcup_{\varphi \in X} \varphi_1(v)) \times \cdots \times (\lambda v. \bigcup_{\varphi \in X} \varphi_k(v))$

Note 15 *Similar results to Note 13 hold for $[Val \rightarrow P]$, $[Val \rightarrow Pow(Traces)]$*

Lemma 16. *$\gg$: $Traces \times [Val \rightarrow Pow(Traces)] \rightarrow Pow(Traces)$ is continuous*

Proof. Show continuity in each argument separately. Over the left argument it is trivial, since *Traces* is a discrete CPO.
Over the right argument:
Let $X \subseteq [Val \rightarrow Pow(Traces)]$ be directed and $s \in Traces$
Proceed by induction on the number of publications in s
If $\bar{P}(s)$,
$\implies s \gg \bigsqcup X = \{s\} = \bigsqcup_{F \in X} (s \gg F)$
If $s \equiv s_1!vs_2$ and $\bar{P}(s_1)$,
$s \gg \bigsqcup X = s_1\tau\ ((s_2 \gg \bigsqcup X) \parallel \bigcup_{F \in X} F(v))$ by Note 15
$= s_1\tau\ ((\bigcup_{F \in X} s_2 \gg F) \parallel \bigcup_{F \in X} F(v))$ by *IH*
$= s_1\tau\ \bigcup_{F \in X} ((s_2 \gg F) \parallel F(v))$ by Lemma 11
$= \bigcup_{F \in X} s_1\tau\ ((s_2 \gg F) \parallel F(v))$
$= \bigsqcup_{F \in X} s \gg F$ □

Corollary 1. *Let* $S \in Pow(Traces)$ *and* $F \in [Val \rightarrow Pow(Traces)]$. *Then,* $\bigcup_{s \in S} s \gg F$ *is continuous*

Lemma 17. *Prefixing* : $Pow(Traces) \rightarrow P$ *is continuous*

Lemma 18. *Removing an event from a trace-set is a continuous operation.*

Note 19 $<_x$: $Traces \times Traces \rightarrow Pow(Traces)$ *is continuous*

Corollary 2. $<_x$: $Pow(Traces) \times Pow(Traces) \rightarrow Pow(Traces)$
is continuous

Note 20 *All the functions proved to be continuous are also monotonic*

Theorem 5. *For all* f, $[\![f]\!]$ *is continuous*

Proof. We know that $[\![f]\!] \in [Fenv \rightarrow [Env \rightarrow P]]$. We will show the continuity of $[(Fenv \times Env) \rightarrow P]$ and this is enough because currying is a continuous operation.
By structural induction on f.
Let X_φ, X_ρ be directed subsets of $Fenv$ and Env respectively.

a) $let(v)$
$\Longrightarrow \quad [\![f]\!](\bigsqcup X_\varphi)(\bigsqcup X_\rho) = \{!v\}_{\mathrm{p}} = \bigsqcup_{\varphi \in X_\varphi} \bigsqcup_{\rho \in X_\rho} [\![let(v)]\!]\varphi\rho$

b) $\mathbf{0}$ or $M(v)$ or $?k$
as above

c) $let(x)$
$[\![let(x)]\!](\bigsqcup X_\varphi)(\bigsqcup X_\rho) = \bigsqcup_{\varphi \in X_\varphi} [\![let(x)]\!]\varphi(\bigsqcup X_\rho)$ $\quad (c1)$
Cases on X_ρ:
  - If $\exists \rho \in X_\rho.\ \rho(x) = \text{Absent}$ then $\forall \rho \in X_\rho.\ \rho(x) = \text{Absent}$ because X_ρ is directed.
    $(c1) \Rightarrow \bigsqcup_{\varphi \in X_\varphi} \{\varepsilon\} = \bigsqcup_{\varphi \in X_\varphi} \bigsqcup_{\rho \in X_\rho} [\![let(x)]\!]\varphi\rho$
  - If $\exists \rho \in X_\rho.\ \rho(x) = v$ then $\forall \rho \in X_\rho.\ \rho(x) = v$ because X_ρ is directed.
    $(c1) \Rightarrow \bigsqcup_{\varphi \in X_\varphi} \{[v/x]\, !v\}_{\mathrm{p}} = \bigsqcup_{\varphi \in X_\varphi} \bigsqcup_{\rho \in X_\rho} [\![let(x)]\!]\varphi\rho$

d) $M(x)$
as above

e) $E_i(v)$
$[\![E_i(v)]\!](\bigsqcup X_\varphi)(\bigsqcup X_\rho) =$
$= \bigsqcup_{\rho \in X_\rho} [\![E_i(v)]\!](\bigsqcup X_\varphi)\rho$
$= \bigsqcup_{\rho \in X_\rho} \{\tau\ t \mid t \in (\bigsqcup X_\varphi)_i(v)\}_{\mathrm{p}}$
$= \bigsqcup_{\rho \in X_\rho} \{\tau\ t \mid t \in \bigcup_{\varphi \in X_\varphi} \varphi_i(v)\}_{\mathrm{p}}$ by Note 15
$= \bigsqcup_{\rho \in X_\rho} \bigcup_{\varphi \in X_\varphi} \{\tau\ t \mid t \in \varphi_i(v)\}_{\mathrm{p}}$ by Lemma 17
$= \bigsqcup_{\varphi \in X_\varphi} \bigsqcup_{\rho \in X_\rho} [\![E_i(v)]\!]\varphi\rho$

f) $E_i(x)$
Cases on X_ρ and similar to the previous case

g) $h \mid g$

$[\![h \mid g]\!](\bigsqcup X_\varphi)(\bigsqcup X_\rho) =$
$= [\![h]\!](\bigsqcup X_\varphi)(\bigsqcup X_\rho) \parallel [\![g]\!](\bigsqcup X_\varphi)(\bigsqcup X_\rho)$
$= (\bigsqcup_{\varphi\in X_\varphi} \bigsqcup_{\rho\in X_\rho} [\![h]\!]\varphi\rho) \parallel (\bigsqcup_{\varphi\in X_\varphi} \bigsqcup_{\rho\in X_\rho} [\![g]\!]\varphi\rho)$ by *IH*
$= \bigsqcup_{\varphi\in X_\varphi} \bigsqcup_{\rho\in X_\rho} ([\![h]\!]\varphi\rho \parallel [\![g]\!]\varphi\rho)$ by Lemma 11
$= \bigsqcup_{\varphi\in X_\varphi} \bigsqcup_{\rho\in X_\rho} [\![h \mid g]\!]\varphi\rho$

h) $h >x> g$

$[\![h >x> g]\!](\bigsqcup X_\varphi)(\bigsqcup X_\rho) =$
$= \bigcup_{s\in[\![h]\!](\bigsqcup X_\varphi)(\bigsqcup X_\rho)} s \gg \lambda v.([\![g]\!](\bigsqcup X_\varphi)(\bigsqcup X_\rho)[x=v])\backslash[v/x]$ $(h1)$
But by *IH*,
$[\![h]\!](\bigsqcup X_\varphi)(\bigsqcup X_\rho) = \bigsqcup_{\varphi\in X_\varphi} \bigsqcup_{\rho\in X_\rho} [\![h]\!]\varphi\rho$ $(h2)$
Also, by *IH* and Lemmas 12, 18 we get
$\lambda v.([\![g]\!](\bigsqcup X_\varphi)(\bigsqcup X_\rho)[x=v])\backslash[v/x] =$
$= \lambda v. \bigsqcup_{\varphi\in X_\varphi} \bigsqcup_{\rho\in X_\rho} ([\![g]\!]\varphi\rho[x=v])\backslash[v/x]$
$= \bigsqcup_{\varphi\in X_\varphi} \bigsqcup_{\rho\in X_\rho} \lambda v.([\![g]\!]\varphi\rho[x=v])\backslash[v/x]$ by Note 15
Using the above and $h2$ and Corollary 1, we get the desired result by $h1$.

i) h **where** $x :\in g$

By Lemma 12 and *IH*,
$[\![h]\!](\bigsqcup X_\varphi)(\bigsqcup X_\rho)[x=v] = \bigsqcup_{\varphi\in X_\varphi} \bigsqcup_{\rho\in X_\rho} [\![h]\!]\varphi\rho[x=v]$
$\Longrightarrow \quad \bigcup_{v\in Val} \bigsqcup_{\varphi\in X_\varphi} \bigsqcup_{\rho\in X_\rho} [\![h]\!]\varphi\rho[x=v] =$
$= \bigsqcup_{\varphi\in X_\varphi} \bigsqcup_{\rho\in X_\rho} \bigcup_{v\in Val} [\![h]\!]\varphi\rho[x=v]$
By this, *IH* for g and Corollary 2 we get the result □

C Prefix-Closure Proofs

Lemma 21. $t_1 \parallel t_2 = t_1 \breve{\parallel} t_2$

Proof. By induction on $|t_1| + |t_2|$.
The only interesting case is when $|t_1| \geq 2$ and $|t_2| \geq 2$ i.e. $t_1 = a_1 t_1' a_2$ and $t_2 = b_1 t_2' b_2$

$\Longrightarrow \quad t_1 \parallel t_2 = a_1(t_1' a_2 \parallel t_2) \cup b_1(t_1 \parallel t_2' b_2)$

$= a_1(t_1' a_2 \breve{\parallel} t_2) \cup b_1(t_1 \breve{\parallel} t_2' b_2)$ by *IH*

$= a_1((t_1' \breve{\parallel} t_2)a_2 \cup (t_1' a_2 \breve{\parallel} b_1 t_2')b_2) \cup b_1((a_1 t_1' \breve{\parallel} t_2' b_2)a_2 \cup (t_1 \breve{\parallel} t_2')b_2)$

$= a_1(t_1' \breve{\parallel} t_2)a_2 \cup a_1(t_1' a_2 \breve{\parallel} b_1 t_2')b_2 \cup b_1(a_1 t_1' \breve{\parallel} t_2' b_2)a_2 \cup b_1(t_1 \breve{\parallel} t_2')b_2$

$= (a_1(t_1' \breve{\parallel} t_2) \cup b_1(a_1 t_1' \breve{\parallel} t_2' b_2))a_2 \cup (a_1(t_1' a_2 \breve{\parallel} b_1 t_2') \cup b_1(t_1 \breve{\parallel} t_2'))b_2$

$= (a_1(t_1' \parallel t_2) \cup b_1(a_1 t_1' \parallel t_2' b_2))a_2 \cup (a_1(t_1' a_2 \parallel b_1 t_2') \cup b_1(t_1 \parallel t_2'))b_2$ by *IH*

$= (a_1 t_1' \parallel t_2)a_2 \cup (t_1 \parallel b_1 t_2')b_2$

$= (a_1 t_1' \breve{\parallel} t_2)a_2 \cup (t_1 \breve{\parallel} b_1 t_2')b_2$ by *IH*

$= t_1 \breve{\parallel} t_2$ □

By this lemma, we can use the operators $\parallel$ and $\breve{\parallel}$ interchangeably.

Lemma 22. *$T_1, T_2 \in P$ implies $T_1 \parallel T_2 \in P$*

Proof. By Lemma 21, suffices to show that $T_1 \breve{\parallel} T_2 \in P$, i.e. suffices to show that for all $t \in T_1 \breve{\parallel} T_2$, $t_\mathrm{p} \subseteq T_1 \breve{\parallel} T_2$
By induction on $|t|$
Since $t \in T_1 \breve{\parallel} T_2$, then $\exists t_1 \in T_1, t_2 \in T_2.\ t \in t_1 \breve{\parallel} t_2$ (1)
The only interesting case is when $|t| \geq 2$ and $t_1 = t_1' a$ and $t_2 = t_2' b$

$\Longrightarrow \quad t \in ((t_1' \breve{\parallel} t_2)a \cup (t_1 \breve{\parallel} t_2')b)$

$\Longrightarrow \quad t_\mathrm{p} \subseteq ((t_1' \breve{\parallel} t_2)a \cup (t_1 \breve{\parallel} t_2')b)_\mathrm{p}$

$\Longrightarrow \quad t_\mathrm{p} \subseteq ((t_1' \breve{\parallel} t_2)_\mathrm{p} \cup (t_1' \breve{\parallel} t_2)a \cup (t_1 \breve{\parallel} t_2')_\mathrm{p} \cup (t_1 \breve{\parallel} t_2')b)$ (2)

But $T_1 \in P \Rightarrow t_1' \in T_1$ and $T_2 \in P \Rightarrow t_2' \in T_2$

$\Longrightarrow$ by *IH*, $(t_1' \breve{\parallel} t_2)_\mathrm{p} \subseteq T_1 \breve{\parallel} T_2$ and $(t_1 \breve{\parallel} t_2')_\mathrm{p} \subseteq T_1 \breve{\parallel} T_2$

$\Longrightarrow$ by 2, suffices to show that $((t_1' \breve{\parallel} t_2)a \cup (t_1 \breve{\parallel} t_2')b) \subseteq T_1 \breve{\parallel} T_2$

i.e. that $t_1 \breve{\parallel} t_2 \subseteq T_1 \breve{\parallel} T_2$ which holds by 1 □

Lemma 23. *If $F \in [Val \to P]$ and $s \in Traces$, then $(\bigcup_{s' \in s_\mathrm{p}} s' \gg F) \in P$*

Proof. By induction on the number of publications in s.
If no publications in s,

$\Longrightarrow \quad \bigcup_{s' \in s_\mathrm{p}} s' \gg F = \bigcup_{s' \in s_\mathrm{p}} \{s'\} = s_\mathrm{p} \in P$

If $s = s_1!vs_2$ and no publications in s_1,

$\Longrightarrow \quad \bigcup_{s' \in s_\mathrm{p}} s' \gg F = (\bigcup_{s' \in (s_1)_\mathrm{p}} s' \gg F) \cup (s_1!v \gg F) \cup (\bigcup_{s' \in s_1!v(s_2)_\mathrm{p}} s' \gg F)$

$= (s_1)_\mathrm{p} \cup \{s_1\tau\} \cup s_1\tau((\bigcup_{s' \in (s_2)_\mathrm{p}} s' \gg F) \parallel F(v))$

$= \{s_1\tau\}_\mathrm{p} \cup s_1\tau((\bigcup_{s' \in (s_2)_\mathrm{p}} s' \gg F) \parallel F(v))$

$\Longrightarrow$ suffices to show that $((\bigcup_{s' \in (s_2)_\mathrm{p}} s' \gg F) \parallel F(v)) \in P$

which, by Lemma 22, follows by $(\bigcup_{s' \in (s_2)_\mathrm{p}} s' \gg F) \in P$ and $F(v) \in P$, which holds by *IH* for s_2

Corollary 3. *If* $T \in P$ *and* $F \in [Val \to P]$, *then* $(\bigcup_{s \in T} s \gg F) \in P$ $\square$

Lemma 24. $T_1, T_2 \in P$ *implies* $T_1 <_x T_2 \in P$

Proof. If $t \in T_1 <_x T_2$ then $\exists t_1 \in T_1, t_2 \in T_2.\, t \in t_1 <_x t_2$
We must show that $t_{\mathrm{p}} \subseteq T_1 <_x T_2$.
Cases depending on which branch of the definition of $<_x$ was used

a) $t \in t_1 \parallel t_2,\ \bar{R}(x, t_1),\ \bar{P}(t_2)$ (1)
$\implies t \in \bigcup_{t_1' \in (t_1)_{\mathrm{p}}, t_2' \in (t_2)_{\mathrm{p}}} t_1' \parallel t_2' = (t_1)_{\mathrm{p}} \parallel (t_2)_{\mathrm{p}}$ by Note 20
$\implies t_{\mathrm{p}} \subseteq ((t_1)_{\mathrm{p}} \parallel (t_2)_{\mathrm{p}})_{\mathrm{p}} = (t_1)_{\mathrm{p}} \parallel (t_2)_{\mathrm{p}}$ by Lemma 22
By 1, $(t_1)_{\mathrm{p}} <_x (t_2)_{\mathrm{p}} = (t_1)_{\mathrm{p}} \parallel (t_2)_{\mathrm{p}}$
$\implies t_{\mathrm{p}} \subseteq (t_1)_{\mathrm{p}} <_x (t_2)_{\mathrm{p}}$
$\implies t_{\mathrm{p}} \subseteq T_1 <_x T_2$ by Note 20

b) $t \in t_1 \parallel t_{21}\tau,\ \bar{R}(x, t_1),\ t_2 = t_{21}!v\, t_{22},\ \bar{P}(t_{21})$
$\implies t \in (t_1)_{\mathrm{p}} \parallel (t_{21}\tau)_{\mathrm{p}}$ by Note 20
$\implies t_{\mathrm{p}} \subseteq ((t_1)_{\mathrm{p}} \parallel (t_{21}\tau)_{\mathrm{p}})_{\mathrm{p}} = (t_1)_{\mathrm{p}} \parallel (t_{21}\tau)_{\mathrm{p}}$ by Lemma 22
$\implies t_{\mathrm{p}} \subseteq ((t_1)_{\mathrm{p}} \parallel (t_{21})_{\mathrm{p}}) \cup ((t_1)_{\mathrm{p}} \parallel \{t_{21}\tau\})$
$\implies t_{\mathrm{p}} \subseteq ((t_1)_{\mathrm{p}} <_x (t_{21})_{\mathrm{p}}) \cup ((t_1)_{\mathrm{p}} <_x \{t_{21}!v\})$
$\implies t_{\mathrm{p}} \subseteq (t_1)_{\mathrm{p}} <_x (t_{21}!v)_{\mathrm{p}}$
$\implies t_{\mathrm{p}} \subseteq T_1 <_x T_2$ by Note 20

c) $t \in (t_{11} \parallel t_{21}\tau)(t_{12} \backslash [v/x])$,
$t_1 = t_{11}[v/x]t_{12},\ \bar{R}(x, t_{11}),\ t_2 = t_{21}!v\, t_{22},\ \bar{P}(t_{21})$
$\implies t_{\mathrm{p}} \in (t_{11} \parallel t_{21}\tau)_{\mathrm{p}} \cup (t_{11} \parallel t_{21}\tau)(t_{12} \backslash [v/x])_{\mathrm{p}}$
$\implies t_{\mathrm{p}} \in (\{t_{11}\}_{\mathrm{p}} \parallel \{t_{21}\tau\}_{\mathrm{p}})_{\mathrm{p}} \cup (t_{11} \parallel t_{21}\tau)(t_{12} \backslash [v/x])_{\mathrm{p}}$ by Note 20
$\implies t_{\mathrm{p}} \in (\{t_{11}\}_{\mathrm{p}} \parallel \{t_{21}\tau\}_{\mathrm{p}}) \cup (t_{11} \parallel t_{21}\tau)(t_{12} \backslash [v/x])_{\mathrm{p}}$ by Lemma 22
By the previous case, this can be written
$\implies t_{\mathrm{p}} \in (\{t_{11}\}_{\mathrm{p}} <_x \{t_{21}!v\}_{\mathrm{p}}) \cup (t_{11}[v/x]\{t_{12}\}_{\mathrm{p}} <_x \{t_{21}!v\})$
$\implies t_{\mathrm{p}} \in (\{t_{11}\}_{\mathrm{p}} <_x \{t_{21}!v\}_{\mathrm{p}}) \cup (t_{11}[v/x]\{t_{12}\}_{\mathrm{p}} <_x \{t_{21}!v\}_{\mathrm{p}})$ by Note 20
$\implies t_{\mathrm{p}} \subseteq \{t_1\}_{\mathrm{p}} <_x \{t_{21}!v\}_{\mathrm{p}}$
$\implies t_{\mathrm{p}} \subseteq T_1 <_x T_2$ by Note 20 $\square$

Theorem 6. *For all* f, $[\![f]\!]\varphi\rho \in P$

Proof. By structural induction on f, using Lemmas 22, 24 and Corollary 3 $\square$

D Denotational Lemmas

Lemma 25. *Let* $t \in [\![f]\!]\varphi\rho$
Then, $[v/x] \hat{\in} t$ *implies* $\rho(x) = v$ *and* $x \in \mathrm{fv}(f)$

Proof. By structural induction on f. The interesting cases are:

a) $f \equiv let(x)$
Clearly, $x \in \mathrm{fv}(f)$.
Also, $[v/x] \hat{\in} t$ implies $t \in \{[v/x]\,!v\}_{\mathrm{p}}$
$\Longrightarrow \quad \rho(x) = v$

b) $f \equiv E_i(v)$
There are no receives in the traces of $\varphi_i(v)$ so this case is vacuously true.

c) $f \equiv h \mid g$
$t \in [\![h \mid g]\!]\varphi\rho$
$\Longrightarrow \quad \exists\, t_1 \in [\![h]\!]\varphi\rho,\ t_2 \in [\![g]\!]\varphi\rho.\ t \in t_1 \parallel t_2$
Therefore, either $[v/x] \hat{\in} t_1$ or $[v/x] \hat{\in} t_2$
- $[v/x] \hat{\in} t_1$
  $\Longrightarrow \quad \rho(x) = v, \quad x \in \mathrm{fv}(h)$ by *IH* for h
  $\Longrightarrow \quad x \in \mathrm{fv}(f)$
- $[v/x] \hat{\in} t_2$ similarly

d) $f \equiv h > y > g, \quad x \neq y$
$t \in [\![h > y > g]\!]\varphi\rho$
$\Longrightarrow \quad \exists\, s \in [\![h]\!]\varphi\rho. \quad t \in s \gg \lambda w.([\![g]\!]\varphi\rho[y = w]) \backslash [w/y]$
- $[v/x]$ is an event of s
  $\Longrightarrow \quad \rho(x) = v, \quad x \in \mathrm{fv}(h)$ by *IH* for h
  $\Longrightarrow \quad x \in \mathrm{fv}(f)$
- $[v/x]$ comes from the traces of $([\![g]\!]\varphi\rho[y = w]) \backslash [w/y]$ for some w
  $\Longrightarrow \quad x \in \mathrm{fv}(g), \quad \rho[y = w](x) = v$ by *IH* for g
  $\Longrightarrow \quad x \in \mathrm{fv}(f), \quad \rho(x) = v$

Similarly if f is $h > x > g$ □

Corollary 4. *If* $t \in [\![f]\!]\varphi\rho$, $[v/x] \hat{\in} t$ *and* $v \neq w$ *then* $[w/x] \hat{\notin} t$

Proof. $[v/x] \hat{\in} t$
$\Longrightarrow \quad \rho(x) = v$ by Lemma 25
$\Longrightarrow \quad \rho(x) \neq w$
$\Longrightarrow \quad [w/x] \hat{\notin} t$ by the contrapositive of Lemma 25 □

Lemma 26. *Let* $t \in [\![f]\!]\varphi\rho$.

Then, $\bar{R}(x, t)$ *implies* $t \in [\![f]\!]\varphi\rho'$ *where* $\rho'(y) = \begin{cases} \rho(y) & x \neq y \\ \textit{anything} & x = y \end{cases}$

Proof. By structural induction on f. The interesting cases are:

a) $f \equiv let(x)$
$\bar{R}(x, t) \quad \Rightarrow \quad t = \varepsilon \quad \Rightarrow \quad t \in [\![f]\!]\varphi\rho'$

b) $f \equiv [\![h \mid g]\!]\varphi\rho$

$\implies \quad \exists\, t_1 \in [\![h]\!]\varphi\rho,\ t_2 \in [\![g]\!]\varphi\rho.\ t \in t_1 \parallel t_2$

We know $\bar{R}(x,t)$

$\implies \quad \bar{R}(x,t_1), \quad \bar{R}(x,t_2)$

$\implies \quad t_1 \in [\![h]\!]\varphi\rho', \quad t_2 \in [\![g]\!]\varphi\rho'$ by IH for h, g

$\implies \quad t \in [\![h \mid g]\!]\varphi\rho'$

c) $f \equiv h$ **where** $x :\in g$

$t \in [\![h$ **where** $x :\in g]\!]\varphi\rho$

$\implies \quad \exists\, t_1 \in \bigcup_{w \in Val}[\![h]\!]\varphi\rho[x=w],\ t_2 \in [\![g]\!]\varphi\rho. \quad t \in t_1 <_x t_2$

We must proceed by cases depending on which branch of the definition of $<_x$ was used. We examine only one case, the others are similar.

Suppose $t_1 = t_{11}[v/x]t_{12},\ \bar{R}(x,t_{11}),\ t_2 = t_{21}!v\, t_{22},\ \bar{P}(t_{21})$

$\implies \quad t \in (t_{11} \parallel t_{21}\tau)(t_{12}\backslash[v/x])$ $(c1)$

$\implies \quad t \in t_1 <_x t_2'$ where $t_2' = t_{21}!v$

It suffices to show that that $t_1 \in \bigcup_{w \in Val}[\![h]\!]\varphi\rho'[x=w], \quad t_2' \in [\![g]\!]\varphi\rho'$

By Corollary 4, $[v'/x] \hat{\notin} t_1$ when $v \neq v'$

Then, by $c1$, $\bar{R}(x,t)$ implies $\bar{R}(x,t_{21})$

$\implies \quad \bar{R}(x,t_2')$ $(c2)$

But by Theorem 6, $t_2' \in [\![g]\!]\varphi\rho$

$\implies \quad t_2' \in [\![g]\!]\varphi\rho'$ by $c2$ and IH for g

Also, by Lemma 25 $\quad t_1 \in [\![h]\!]\varphi\rho[x=v]$

$\implies \quad t_1 \in [\![h]\!]\varphi\rho'[x=v]$

$\implies \quad t_1 \in \bigcup_{w \in Val}[\![h]\!]\varphi\rho'[x=w]$

Similarly when f is $(h$ **where** $y :\in g)$ and $x \neq y$ □

Corollary 5 (Weakening).
If $x \notin \text{fv}(f)$ *then* $[\![f]\!]\varphi\rho = [\![f]\!]\varphi\rho[x=v]$ *for any* v

Proof. Let $t \in [\![f]\!]\varphi\rho$ and $x \notin \text{fv}(f)$

$\implies \quad \bar{R}(x,t)$ by the contrapositive of Lemma 25

$\implies \quad t \in [\![f]\!]\varphi\rho[x=v]$ for any v by Lemma 26

$\implies \quad [\![f]\!]\varphi\rho \subseteq [\![f]\!]\varphi\rho[x=v]$

Similarly, $[\![f]\!]\varphi\rho[x=v] \subseteq [\![f]\!]\varphi\rho$ □

Corollary 6. $[\![f]\!]\varphi\rho_{-x} \subseteq [\![f]\!]\varphi\rho[x=v]$ *for any* v

Proof. $t \in [\![f]\!]\varphi\rho_{-x}$

$\implies \quad \bar{R}(x,t)$ by the contrapositive of Lemma 25

$\implies \quad t \in [\![f]\!]\varphi\rho[x=v]$ for any v by Lemma 26

$\implies \quad [\![f]\!]\varphi\rho_{-x} \subseteq [\![f]\!]\varphi\rho[x=v]$ □

Lemma 27. $(t_1 \parallel t_2)\backslash a = t_1\backslash a \parallel t_2\backslash a$

Proof. By induction on $|t_1| + |t_2|$

The interesting case is when $|t_1| + |t_2| \geq 2$ and $t_1 = bt_1',\ t_2 = ct_2'$

Then, $(t_1 \parallel t_2)\backslash a = (b(t_1' \parallel t_2) \cup c(t_1 \parallel t_2'))\backslash a$

$= (b(t_1' \parallel t_2))\backslash a \cup (c(t_1 \parallel t_2'))\backslash a$

If $b \neq a$ and $c \neq a$ the above becomes

$= b(t'_1 \parallel t_2)\backslash a \cup c(t_1 \parallel t'_2)\backslash a$
$= b(t'_1\backslash a \parallel t_2\backslash a) \cup c(t_1\backslash a \parallel t'_2\backslash a)$ by *IH*
$= t_1\backslash a \parallel t_2\backslash a$
Similarly when b and/or c is equal to a □

Corollary 7. $(T_1 \parallel T_2)\backslash a = T_1\backslash a \parallel T_2\backslash a$ □

Lemma 28. *Let $s \in Traces$, $x \in Var$, $v \in Val$ and $F : Val \to Pow(Traces)$. Then, $(s \gg F)\backslash[v/x] = s\backslash[v/x] \gg \lambda w.F(w)\backslash[v/x]$*

Proof. By induction on the number of publications in s
If $\bar{P}(s)$ then $(s \gg F)\backslash[v/x] = \{s\}\backslash[v/x] = s\backslash[v/x] \gg \lambda w.F(w)\backslash[v/x]$
If $s = s_1!us_2$ and $\bar{P}(s_1)$ then
$(s \gg F)\backslash[v/x] = (s_1\tau)\backslash[v/x]((s_2 \gg F) \parallel F(u))\backslash[v/x]$
$= (s_1\tau)\backslash[v/x]((s_2 \gg F)\backslash[v/x] \parallel F(u)\backslash[v/x])$ by Corollary 7
$= (s_1\tau)\backslash[v/x]((s_2\backslash[v/x] \gg \lambda w.F(w)\backslash[v/x]) \parallel F(u)\backslash[v/x])$ by *IH* for s_2
$= (s_1!us_2)\backslash[v/x] \gg \lambda w.F(w)\backslash[v/x]$
$= s\backslash[v/x] \gg \lambda w.F(w)\backslash[v/x]$ □

Lemma 29. $(t_1 <_y t_2)\backslash[v/x] = t_1\backslash[v/x] <_y t_2\backslash[v/x]$, *when* $y \neq x$ *and* $(t_1 <_x t_2)\backslash[v/x] = t_1 <_x t_2\backslash[v/x]$

Proof. Assume a well-formedness constraint for t_1, t_2 similar to Corollary 4. Cases depending on which branch of the definition of $<_x$ was used:

a) $\bar{R}(y, t_1)$, $\bar{P}(t_2)$, $t_1 <_y t_2 = t_1 \parallel t_2$
$\Longrightarrow$ holds by Lemma 27
b) $\bar{R}(y, t_1)$, $t_2 = t_{21}!w\, t_{22}$, $\bar{P}(t_{21})$ $t_1 <_y t_2 = t_1 \parallel t_{21}\,\tau$
$\Longrightarrow$ holds by Lemma 27
c) $t_1 = t_{11}[w/y]\, t_{12}$, $\bar{R}(y, t_{11})$, $t_2 = t_{21}!w\, t_{22}$, $\bar{P}(t_{21})$,
$t_1 <_y t_2 = (t_{11} \parallel t_{21}\,\tau)(t_{12}\backslash[w/y])$ $(c1)$
When $x \neq y$, by $c1 \Rightarrow ((t_{11} \parallel t_{21}\,\tau)(t_{12}\backslash[w/y]))\backslash[v/x]$
$= (t_{11} \parallel t_{21}\,\tau)\backslash[v/x]\; (t_{12}\backslash[w/y])\backslash[v/x]$
$= (t_{11}\backslash[v/x] \parallel (t_{21}\,\tau)\backslash[v/x])\; (t_{12}\backslash[w/y])\backslash[v/x]$ by Corollary 7
$= t_1\backslash[v/x] <_y t_2\backslash[v/x]$
When $x = y$, by $c1 \Rightarrow ((t_{11} \parallel t_{21}\,\tau)(t_{12}\backslash[w/x]))\backslash[v/x]$
$= (t_{11} \parallel t_{21}\,\tau)\backslash[v/x]\; (t_{12}\backslash[w/x])\backslash[v/x]$ $(c2)$
By the well-formedness constraint, $[v/x] \hat{\notin} t_{12}$ when $v \neq w$,
therefore $(t_{12}\backslash[w/x])\backslash[v/x] = t_{12}\backslash[w/x]$
$(c2) \Rightarrow (t_{11} \parallel (t_{21}\,\tau)\backslash[v/x])\; (t_{12}\backslash[w/x])$
$= t_1 <_x t_2\backslash[v/x]$ □

Lemma 30 (Substitution). $[\![[v/x]f]\!]\varphi\rho = ([\![f]\!]\varphi\rho[x = v])\backslash[v/x]$

Proof. By structural induction on f.

a) f is $let(x)$ or $let(v)$ or $M(x)$ ...
by inspection of the trace definitions

b) $f \equiv h \mid g$

$[\![[v/x]h \mid [v/x]g]\!]\varphi\rho = [\![[v/x]h]\!]\varphi\rho \parallel [\![[v/x]g]\!]\varphi\rho$

$= ([\![h]\!]\varphi\rho[x=v])\backslash[v/x] \parallel ([\![g]\!]\varphi\rho[x=v])\backslash[v/x]$ by IH

$= ([\![h]\!]\varphi\rho[x=v] \parallel [\![g]\!]\varphi\rho[x=v])\backslash[v/x]$ by Corollary 7

$= ([\![h \mid g]\!]\varphi\rho[x=v])\backslash[v/x]$

c) $f \equiv h > x > g$ (Similarly when $f \equiv h > y > g, x \neq y$)

$[v/x]f \equiv [v/x]h > x > g$, so

$[\![[v/x]h > x > g]\!]\varphi\rho = \bigcup_{s \in [\![[v/x]h]\!]\varphi\rho} s \gg \lambda w.([\![g]\!]\varphi\rho[x=w])\backslash[w/x]$

$= \bigcup_{s \in ([\![h]\!]\varphi\rho[x=v])\backslash[v/x]} s \gg \lambda w.([\![g]\!]\varphi\rho[x=w])\backslash[w/x]$ by IH $(c1)$

By Lemma 25, if $v \neq w$ then $[v/x]$ is not in the traces of $[\![g]\!]\varphi\rho[x=w]$

$\Longrightarrow \quad ([\![g]\!]\varphi\rho[x=w])\backslash[w/x] = ([\![g]\!]\varphi\rho[x=w])\backslash[w/x]\backslash[v/x]$

$= ([\![g]\!]\varphi\rho[x=v][x=w])\backslash[w/x]\backslash[v/x]$

The above also holds trivially if $v = w$, so

$c1 \Rightarrow \bigcup_{s \in [\![h]\!]\varphi\rho[x=v]} s\backslash[v/x] \gg \lambda w.([\![g]\!]\varphi\rho[x=v][x=w])\backslash[w/x]\backslash[v/x]$

$= \bigcup_{s \in [\![h]\!]\varphi\rho[x=v]} (s \gg \lambda w.([\![g]\!]\varphi\rho[x=v][x=w])\backslash[w/x])\backslash[v/x]$ by Lem. 28

$= ([\![h > x > g]\!]\varphi\rho[x=v])\backslash[v/x]$

d) $f \equiv h$ **where** $x :\in g$ (Similarly when $f \equiv h$ **where** $y :\in g, x \neq y$)

$[v/x]f \equiv h$ **where** $x :\in [v/x]g$, so

$[\![h$ **where** $x :\in [v/x]g]\!]\varphi\rho =$

$= \bigcup_{w \in Val} [\![h]\!]\varphi\rho[x=w] <_x [\![[v/x]g]\!]\varphi\rho$

$= \bigcup_{w \in Val} [\![h]\!]\varphi\rho[x=w] <_x ([\![g]\!]\varphi\rho[x=v])\backslash[v/x]$ by IH

Let $T_1 = \bigcup_{w \in Val} [\![h]\!]\varphi\rho[x=w]$, $T_2 = [\![g]\!]\varphi\rho[x=v]$

then the above becomes $T_1 <_x T_2\backslash[v/x]$

$= \bigcup_{t_1 \in T_1, t_2 \in T_2\backslash[v/x]} t_1 <_x t_2$

$= \bigcup_{t_1 \in T_1, t_2 \in T_2} t_1 <_x t_2\backslash[v/x]$

$= \bigcup_{t_1 \in T_1, t_2 \in T_2} (t_1 <_x t_2)\backslash[v/x]$ by Lemma 29

$= (\bigcup_{t_1 \in T_1, t_2 \in T_2} t_1 <_x t_2)\backslash[v/x]$

$= (T_1 <_x T_2)\backslash[v/x]$

$= (\bigcup_{w \in Val} [\![h]\!]\varphi\rho[x=w] <_x [\![g]\!]\varphi\rho[x=v])\backslash[v/x]$

$= ([\![h$ **where** $x :\in g]\!]\varphi\rho[x=v])\backslash[v/x]$ □

E Operational Lemmas

All Lemmas in this section are proved by induction on the height of the derivation and the proofs are straightforward.

Lemma 31 (Take a receive step). *If* $\Delta, \Gamma \vdash f \stackrel{[v/x]}{\rightarrow} f'$ *then*

1. $x \in \mathrm{fv}(f)$
2. $\Gamma(x) = v$
3. $[v/x]f' \equiv [v/x]f$

Lemma 32 (Take a non-receive step). *If* $\Delta, \Gamma \vdash f \stackrel{a}{\rightarrow} f'$ *and* $\bar{R}(x, a)$ *then*

1. $\Delta, \Gamma \vdash [v/x]f \stackrel{a}{\rightarrow} [v/x]f'$ *for any* v
2. $\Delta, \Gamma' \vdash f \stackrel{a}{\rightarrow} f'$ *where* $\Gamma'(y) = \begin{cases} \Gamma(y) & x \neq y \\ \textit{unspecified/anything} & x = y \end{cases}$

Lemma 33. $\Delta, \Gamma \vdash f \stackrel{a}{\rightarrow} f'$ *implies* $\mathrm{fv}(f') \subseteq \mathrm{fv}(f)$

Proof. By induction on the height of the derivation. The interesting cases are

– (DEF) $\dfrac{}{\Delta, \Gamma \vdash E_i(v) \stackrel{\tau}{\rightarrow} [v/x]f_i} \; (E_i(x) = f_i) \in \Delta$

  $\mathrm{fv}(E_i(v)) = \emptyset = \mathrm{fv}([v/x]f_i)$ by the constraint $\mathrm{fv}(f_i) \subseteq \{x\}$

– (ASYM-L) $\dfrac{\Delta, \Gamma \vdash h \stackrel{a}{\rightarrow} h'}{\Delta, \Gamma \vdash h \textbf{ where } x :\in g \stackrel{a}{\rightarrow} h' \textbf{ where } x :\in g} \; a \neq [v/x]$

  $\mathrm{fv}(h') \subseteq \mathrm{fv}(h)$ by *IH* (I)
  $\mathrm{fv}(h' \textbf{ where } x :\in g) = (\mathrm{fv}(h') - \{x\}) \cup \mathrm{fv}(g)$
  $\subseteq (\mathrm{fv}(h) - \{x\}) \cup \mathrm{fv}(g)$ by I
  $= \mathrm{fv}(h \textbf{ where } x :\in g)$ □

Lemma 34. *If* $\Delta, \Gamma \vdash [v/x]f \stackrel{a}{\rightarrow} f'$ *then*

a) either $\Delta, \Gamma \vdash f \stackrel{a}{\rightarrow} f''$ *where* $[v/x]f'' \equiv f'$
b) or $\Delta, \Gamma[x = v] \vdash f \stackrel{[v/x]}{\rightarrow} f_1 \stackrel{a}{\rightarrow} f_2$ *where* $[v/x]f \equiv [v/x]f_1$, $[v/x]f_2 \equiv f'$

F Soundness - Adequacy

The following lemma is the key lemma for proving soundness. The soundness theorem is an easy corollary of this lemma.

Lemma 35. *If* $\Delta, \Gamma \vdash f \xrightarrow{a} f'$ *and* $t \in [\![f']\!][\![\Delta]\!]\rho$ *then* $at \in [\![f]\!][\![\Delta]\!]\rho$ *where* $\rho = \rho_0[x_1 = v_1] \ldots [x_m = v_m]$ *and* $\Gamma = \{(x_1, v_1), \ldots, (x_m, v_m)\}$.

Proof. Since Γ is a partial function from Var to Val we can assume that the x_i's are pairwise distinct. We proceed by structural induction on f and cases on the reduction rule used

a) (SITEC) $\dfrac{}{\Delta, \Gamma \vdash\ M(v) \xrightarrow{M_k(v)} ?k}$ k fresh
   $[\![?k]\!][\![\Delta]\!]\rho = \{\, \tau\, !w \mid w \in Val\}_{\mathrm{p}}$
   Consider only the case when $t = \tau\, !w$
   Then, $(M_k(v)\, \tau\, !w) \in [\![M(v)]\!][\![\Delta]\!]\rho$

b) (SITEC-VAR) $\dfrac{}{\Delta, \Gamma \vdash\ M(x) \xrightarrow{[v/x]} M(v)}$ $\Gamma(x) = v$
   $[\![M(v)]\!][\![\Delta]\!]\rho = \{\, M_k(v)\, \tau\, !w \mid w \in Val\,, k \text{ fresh}\}_{\mathrm{p}}$
   Consider only the case when $t = M_k(v)\, \tau$
   We know $\Gamma(x) = v$, therefore $\rho(x) = v$
   $\Longrightarrow \quad ([v/x]\ M_k(v)\, \tau) \in [\![M(x)]\!][\![\Delta]\!]\rho$ when $\rho(x) = v$

c) SITERET, LET, LET-VAR, DEF-VAR similarly

d) (DEF) $\dfrac{}{\Delta, \Gamma \vdash\ E_i(v) \xrightarrow{\tau} [v/x]f_i}$ $(E_i(x) \triangleq f_i) \in \Delta$
   Let $t \in [\![[v/x]f_i]\!][\![\Delta]\!]\rho \overset{\text{Lem. 30}}{\Longrightarrow} t \in ([\![f_i]\!][\![\Delta]\!]\rho[x = v]) \backslash [v/x]$ $\quad (d1)$
   Also, $[\![E_i(v)]\!][\![\Delta]\!]\rho = \{\, \tau\, t' \mid t' \in [\![\Delta]\!]_i(v)\}_{\mathrm{p}}$
   where $[\![\Delta]\!]_i(v) = ([\![f_i]\!][\![\Delta]\!]\rho_0[x = v]) \backslash [v/x]$ by $\hat{\Delta}([\![\Delta]\!]) = [\![\Delta]\!]$ $\quad (d2)$
   By $d2$, it suffices to show that $t \in ([\![f_i]\!][\![\Delta]\!]\rho_0[x = v]) \backslash [v/x]$, which holds by $d1$ and Corollary 5, because $x_1, \ldots, x_m$ are not free in f_i

e) (SYM-L) $\dfrac{\Delta, \Gamma \vdash\ h \xrightarrow{a} h'}{\Delta, \Gamma \vdash\ h \mid g \xrightarrow{a} h' \mid g}$
   Let $t \in [\![h' \mid g]\!][\![\Delta]\!]\rho$, then there exist $t_1 \in [\![h']\!][\![\Delta]\!]\rho,\ t_2 \in [\![g]\!][\![\Delta]\!]\rho$
   such that $t \in t_1 \parallel t_2$ $\quad (e1)$
   By IH for h, $at_1 \in [\![h]\!][\![\Delta]\!]\rho \overset{e1}{\Longrightarrow} at \in at_1 \parallel t_2$
   $\Longrightarrow \quad at \in [\![h \mid g]\!][\![\Delta]\!]\rho$

f) Similarly for (SYM-R)

g) (ASYM-L) $\dfrac{\Delta, \Gamma \vdash\ h \xrightarrow{a} h'}{\Delta, \Gamma \vdash\ h \ \textbf{where}\ x :\in g \xrightarrow{a} h' \ \textbf{where}\ x :\in g}$ $a \neq [v/x]$
   Let $t \in [\![h' \ \textbf{where}\ x :\in g]\!][\![\Delta]\!]\rho$, then there exist
   $t_1 \in \bigcup_{v \in Val} [\![h']\!][\![\Delta]\!]\rho[x = v], t_2 \in [\![g]\!][\![\Delta]\!]\rho$ such that $t \in t_1 <_x t_2$ $\quad (g1)$
   Also, by Lemma 32, $\quad \Delta, \Gamma[x = w] \vdash\ h \xrightarrow{a} h'$ $\quad$ for any w $\quad (g2)$
   Cases depending on which branch of the definition of $<_x$ was used for t:

- 1^{st} branch was used,
  $\Longrightarrow\ \bar{R}(x,t_1),\ \bar{P}(t_2),\ t \in t_1 \parallel t_2$ $(g3)$
  By $g1, g2$ and *IH* for h we get $at_1 \in \bigcup_{v\in Val}[\![h]\!][\![\Delta]\!]\rho[x=v]$ $(g4)$
  $\Longrightarrow\ at \in at_1 \parallel t_2$ by $g3$
  $\Longrightarrow\ at \in [\![h \textbf{ where } x :\in g]\!][\![\Delta]\!]\rho$ by $g1, g4$
- 2^{nd} branch was used,
  $\Longrightarrow\ \bar{R}(x,t_1),\ t_2 = t_{21}!u\, t_{22},\ \bar{P}(t_{21}),$
  $t \in t_1 \parallel t_{21}\tau$ $(g5)$
  By $g1, g2$ and *IH* for h we get $at_1 \in \bigcup_{v\in Val}[\![h]\!][\![\Delta]\!]\rho[x=v]$ $(g6)$
  $\Longrightarrow\ at \in at_1 <_x t_2$ by $g5$
  $\Longrightarrow\ at \in [\![h \textbf{ where } x :\in g]\!][\![\Delta]\!]\rho$ by $g1, g6$
- 3^{rd} branch was used,
  $\Longrightarrow\ t_1 = t_{11}[u/x]\, t_{12},\ \bar{R}(x,t_{11}),$
  $t_2 = t_{21}!u\, t_{22},\ \bar{P}(t_{21}),\ t \in (t_{11} \parallel t_{21}\tau)(t_{12}\backslash[u/x])$ $(g7)$
  $t_1 \in [\![h']\!][\![\Delta]\!]\rho[x=u]$ by Lemma 25
  $\Longrightarrow$ by $g2$ and *IH* for h we get $at_1 \in [\![h]\!][\![\Delta]\!]\rho[x=u]$
  $\Longrightarrow\ at_1 \in \bigcup_{v\in Val}[\![h]\!][\![\Delta]\!]\rho[x=v]$ $(g8)$
  $\Longrightarrow\ at \in at_1 <_x t_2$ by $g7$
  $\Longrightarrow\ at \in [\![h \textbf{ where } x :\in g]\!][\![\Delta]\!]\rho$ by $g1, g8$
- 4^{th} branch was used,
  This case is impossible because $t \notin \emptyset$

h) (ASYM-R) Similar to the previous case

i) (ASYM-P) $\dfrac{\Delta, \Gamma \vdash\ g \xrightarrow{!v} g'}{\Delta, \Gamma \vdash\ h \textbf{ where } x :\in g \xrightarrow{\tau} [v/x]h}$

Let $t \in [\![[v/x]h]\!][\![\Delta]\!]\rho$
$\Longrightarrow\ \exists\, t' \in [\![h]\!][\![\Delta]\!]\rho[x=v].\ t = t'\backslash[v/x]$ by Lemma 30 $(i1)$
$\varepsilon \in [\![g']\!][\![\Delta]\!]\rho$ by Thm. 6
$\Longrightarrow\ !v \in [\![g]\!][\![\Delta]\!]\rho$ by *IH* $(i2)$

- $\bar{R}(x,t')$
  $\Longrightarrow\ t = t'$ and $\tau\, t \in t <_x !v$
  $\Longrightarrow\ \tau\, t \in [\![h \textbf{ where } x :\in g]\!][\![\Delta]\!]\rho$ by $i1, i2$
- $t' = t'_1[v/x]t'_2,\ \bar{R}(x,t'_1)$ $(i3)$
  $\Longrightarrow\ \tau\, t'_1(t'_2\backslash[v/x]) \in t' <_x !v$
  $\Longrightarrow\ \tau\, t \in t' <_x !v$ by $i1, i3$
  $\Longrightarrow\ \tau\, t \in [\![h \textbf{ where } x :\in g]\!][\![\Delta]\!]\rho$ by $i1, i2$

j) (SEQ) $\dfrac{\Delta, \Gamma \vdash\ h \xrightarrow{a} h'}{\Delta, \Gamma \vdash\ h >x> g \xrightarrow{a} h' >x> g}\ a \neq !v$

Let $t \in [\![h' >x> g]\!][\![\Delta]\!]\rho$, then there exists $s \in [\![h']\!][\![\Delta]\!]\rho$
such that $t \in s \gg \lambda v.([\![g]\!][\![\Delta]\!]\rho[x=v])\backslash[v/x]$ $(j1)$
Cases on s:

- $\bar{P}(s) \Rightarrow t \in \{s\} \Rightarrow t = s$ $(j2)$
  By *IH* for h, $as \in [\![h]\!][\![\Delta]\!]\rho$
  $\Longrightarrow\ at \in as \gg \lambda v.([\![g]\!][\![\Delta]\!]\rho[x=v])\backslash[v/x]$ by $j2$
  $\Longrightarrow\ at \in [\![h >x> g]\!][\![\Delta]\!]\rho$

- $s = s_1!u\, s_2,\;\; \bar{P}(s_1)$
  Then by $j1$ we get,
  $t \in s_1\tau((s_2 \gg \lambda v.([\![g]\!][\![\Delta]\!]\rho[x = v])\backslash[v/x]) \parallel ([\![g]\!][\![\Delta]\!]\rho[x = u])\backslash[u/x])$
  $\Longrightarrow \quad at \in as \gg \lambda v.([\![g]\!][\![\Delta]\!]\rho[x = v])\backslash[v/x]$ $\quad (j3)$
  By IH for h, $as \in [\![h]\!][\![\Delta]\!]\rho$
  $\Longrightarrow \quad at \in [\![h > x > g]\!][\![\Delta]\!]\rho$ $\quad$ by $j3$

k) (SEQ-P) $\dfrac{\Delta, \Gamma \vdash\; h \xrightarrow{!u} h'}{\Delta, \Gamma \vdash\; h > x > g \xrightarrow{\tau} (h' > x > g) \mid [u/x]g}$

Let $t \in [\![(h' > x > g) \mid [u/x]g]\!][\![\Delta]\!]\rho$, then there exist
$t_1 \in [\![h' > x > g]\!][\![\Delta]\!]\rho,\; t_2 \in [\![[u/x]g]\!][\![\Delta]\!]\rho$ such that $t \in t_1 \parallel t_2$ $\quad (k1)$
By Lemma 30, $t_2 \in ([\![g]\!][\![\Delta]\!]\rho[x = u])\backslash[u/x]$ $\quad (k2)$
By $k1$, $\exists s \in [\![h']\!][\![\Delta]\!]\rho.\; t_1 \in s \gg \lambda v.([\![g]\!][\![\Delta]\!]\rho[x = v])\backslash[v/x]$ $\quad (k3)$
By IH for h, $!u\, s \in [\![h]\!][\![\Delta]\!]\rho$ $\quad (k4)$
By $k1, k2, k3$ $\quad t \in (s \gg \lambda v.([\![g]\!][\![\Delta]\!]\rho[x = v])\backslash[v/x]) \parallel ([\![g]\!][\![\Delta]\!]\rho[x = u])\backslash[u/x]$
$\Longrightarrow \quad \tau\, t \in \tau((s \gg \lambda v.([\![g]\!][\![\Delta]\!]\rho[x = v])\backslash[v/x]) \parallel ([\![g]\!][\![\Delta]\!]\rho[x = u])\backslash[u/x])$
$\Longrightarrow \quad \tau\, t \in !u\, s \gg \lambda v.([\![g]\!][\![\Delta]\!]\rho[x = v])\backslash[v/x]$
$\Longrightarrow \quad \tau\, t \in [\![h > x > g]\!][\![\Delta]\!]\rho$ $\quad$ by $k4$ $\quad \square$

Lemma 36. *If* $\Delta, \Gamma \vdash\; f \xrightarrow{t_1}{}^{*} f'$ *and* $t_2 \in [\![f']\!][\![\Delta]\!]\rho$ *then* $t_1\, t_2 \in [\![f]\!][\![\Delta]\!]\rho$ *where* $\rho = \rho_0[x_1 = v_1]\ldots[x_m = v_m]$ *and* $\Gamma = \{(x_1, v_1), \ldots, (x_m, v_m)\}$, *$x_i$'s are pairwise distinct.*

Proof. By induction on $|t_1|$ $\quad \square$

Theorem 7 (Soundness). *If* $\Gamma = \{(x_1, v_1), \ldots, (x_m, v_m)\}$, $\rho = \rho_0[x_1 = v_1]\ldots[x_m = v_m]$, *x's are pairwise distinct, then*

$$\Delta, \Gamma \vdash\; f \xrightarrow{t}{}^{*} f' \quad \textit{implies} \quad t \in [\![f]\!][\![\Delta]\!]\rho$$

Proof. By induction on $|t|$

- If $|t| = 0 \Leftrightarrow t = \varepsilon$
  $\Longrightarrow \quad \varepsilon \in [\![f]\!][\![\Delta]\!]\rho$ $\quad$ by Thm. 6
- If $t = a\, t'$
  $\Longrightarrow \quad \Delta, \Gamma \vdash\; f \xrightarrow{a} f'' \xrightarrow{t'}{}^{*} f'$
  By IH for t', $t' \in [\![f'']\!][\![\Delta]\!]\rho$ therefore $t \in [\![f]\!][\![\Delta]\!]\rho$ by Lemma 35 $\quad \square$

The following is the key lemma for proving adequacy. Observe that it is the converse of the soundness lemma.

Lemma 37. *If* $at \in [\![f]\!][\![\Delta]\!]\rho$ *then* $\Delta, \Gamma \vdash\; f \xrightarrow{a} f'$ *and* $t \in [\![f']\!][\![\Delta]\!]\rho$ *where* $\rho = \rho_0[x_1 = v_1]\ldots[x_m = v_m]$ *and* $\Gamma = \{(x_1, v_1), \ldots, (x_m, v_m)\}$

Proof. By structural induction on f

a) $f \equiv 0$ $\quad$ vacuously true

b) $f \equiv let(v)$
$\implies$ $[\![let(v)]\!][\![\Delta]\!]\rho = \{!v\}_{\mathrm{p}}$
$\implies$ $a = !v$ and $t = \varepsilon$
Also, $\Delta, \Gamma \vdash\ let(v) \xrightarrow{!v} \mathbf{0}$ and $\varepsilon \in [\![\mathbf{0}]\!][\![\Delta]\!]\rho$

c) $f \equiv M(v)$ or $?k$ similarly

d) $f \equiv let(x)$
For a non-empty trace of f, we know $\rho(x) = v$
$\implies$ $[\![let(x)]\!][\![\Delta]\!]\rho = \{[v/x]\,!v\}_{\mathrm{p}}$
Consider only the case when $a = [v/x]$ and $t = !v$
Then, by LET-VAR, $\Delta, \Gamma \vdash\ let(x) \xrightarrow{[v/x]} let(v)$ and $!v \in [\![let(v)]\!][\![\Delta]\!]\rho$

e) $f \equiv M(x)$ similarly

f) $f \equiv E_i(v)$
$\implies$ $[\![E_i(v)]\!][\![\Delta]\!]\rho = \{\, \tau\, t \mid t \in [\![\Delta]\!]_i(v)\}_{\mathrm{p}}$
By DEF, $\Delta, \Gamma \vdash\ E_i(v) \xrightarrow{\tau} [v/x]f_i$
$\implies$ suffices to show that for any $t \in [\![\Delta]\!]_i(v)$ then $t \in [\![[v/x]f_i]\!][\![\Delta]\!]\rho$
We know $[\![\Delta]\!] = \mathrm{fix}(\hat{\Delta}) \Rightarrow \hat{\Delta}([\![\Delta]\!]) = [\![\Delta]\!]$
Then, $t \in [\![\Delta]\!]_i(v)$ implies $t \in ([\![f_i]\!][\![\Delta]\!]\rho_0[x = v])\backslash[v/x]$
$\implies$ $t \in [\![[v/x]f_i]\!][\![\Delta]\!]\rho_0$ by Lemma 30
$\implies$ $t \in [\![[v/x]f_i]\!][\![\Delta]\!]\rho$ by Corollary 5 because $\mathrm{fv}([v/x]f_i) = \emptyset$

g) $f \equiv h \mid g$
Let $a\,t \in [\![h \mid g]\!][\![\Delta]\!]\rho$, then there exist
$t_1 \in [\![h]\!][\![\Delta]\!]\rho, t_2 \in [\![g]\!][\![\Delta]\!]\rho$ such that $a\,t \in t_1 \parallel t_2$ $(g1)$

- 'a' is an event of t_1, i.e. $t_1 = a\,t_1'$, and by $g1$, $t \in t_1' \parallel t_2$ $(g2)$
  By *IH* for h, $\Delta, \Gamma \vdash\ h \xrightarrow{a} h'$ and $t_1' \in [\![h']\!][\![\Delta]\!]\rho$ $(g3)$
  $\implies$ $\Delta, \Gamma \vdash\ h \mid g \xrightarrow{a} h' \mid g$ by SYM-L
  $\implies$ $t \in [\![h' \mid g]\!][\![\Delta]\!]\rho$ by $g1, g2, g3$
- 'a' is an event of t_2, similarly

h) $f \equiv h >x> g$
Let $a\,t \in [\![h >x> g]\!][\![\Delta]\!]\rho$ then there exists
$s \in [\![h]\!][\![\Delta]\!]\rho$ such that $a\,t \in s \gg \lambda w.([\![g]\!][\![\Delta]\!]\rho[x = w])\backslash[w/x]$ $(h1)$

- $\bar{P}(s)$
  $\implies$ $a\,t = s$ by $h1$
  $\implies$ $\Delta, \Gamma \vdash\ h \xrightarrow{a} h'$ and $t \in [\![h']\!][\![\Delta]\!]\rho$ by *IH* for h $(h2)$
  $\implies$ $\Delta, \Gamma \vdash\ h >x> g \xrightarrow{a} h' >x> g$ by SEQ
  $\implies$ suffices to show that $t \in [\![h' >x> g]\!][\![\Delta]\!]\rho$ which holds by $h2$
- $s = s_1\,!v\,s_2$, $\bar{P}(s_1)$
  Then, by $h1$
  $a\,t \in s_1\tau\ \ ((s_2 \gg \lambda w.([\![g]\!][\![\Delta]\!]\rho[x = w])\backslash[w/x]) \parallel$
  $([\![g]\!][\![\Delta]\!]\rho[x = v])\backslash[v/x])$ $(h3)$
  - 'a' is the first event of s_1, $s_1 = a\,s_1'$
    $\implies$ $\Delta, \Gamma \vdash\ h \xrightarrow{a} h'$ and $s_1'!vs_2 \in [\![h']\!][\![\Delta]\!]\rho$ by *IH* for h $(h4)$
    $\implies$ $\Delta, \Gamma \vdash\ h >x> g \xrightarrow{a} h' >x> g$ by SEQ
    We know that, $t \in s_1'!vs_2 \gg \lambda w.([\![g]\!][\![\Delta]\!]\rho[x = w])\backslash[w/x]$ by $h3$
    $\implies$ $t \in [\![h' >x> g]\!][\![\Delta]\!]\rho$ by $h4$

* s_1 is empty, therefore $s = !v\, s_2$ and by $h3$ $\quad a = \tau$

  $\Longrightarrow \quad \Delta, \Gamma \vdash\ h \xrightarrow{!v} h'$ and $s_2 \in [\![h']\!][\![\Delta]\!]\rho$ by *IH* for h $\quad (h6)$

  Then, by SEQ-P

  $\Delta, \Gamma \vdash\ h >x> g \xrightarrow{\tau} (h' >x> g) \mid [v/x]g$

  By $h3$,

  $t \in (s_2 \gg \lambda w.([\![g]\!][\![\Delta]\!]\rho[x = w])\backslash[w/x]) \parallel ([\![g]\!][\![\Delta]\!]\rho[x = v])\backslash[v/x]$

  $\Longrightarrow \quad t \in [\![h' >x> g]\!][\![\Delta]\!]\rho \parallel ([\![g]\!][\![\Delta]\!]\rho[x = v])\backslash[v/x]$ by $h6$

  $\Longrightarrow \quad t \in [\![(h' >x> g) \mid [v/x]g]\!][\![\Delta]\!]\rho$ by Lemma 30

i) $f \equiv h$ **where** $x :\in g$

Let $a\,t \in [\![h$ **where** $x :\in g]\!][\![\Delta]\!]\rho$, then there exist

$t_1 \in \bigcup_{v \in Val} [\![h]\!][\![\Delta]\!]\rho[x = v], t_2 \in [\![g]\!][\![\Delta]\!]\rho$ such that $a\,t \in t_1 <_x t_2$ $\quad (i1)$

Cases on the branch of the definition of $<_x$ used for $a\,t$

- $\bar{R}(x, t_1),\ \bar{P}(t_2) \Longrightarrow a\,t \in t_1 \parallel t_2$ $\quad (i2)$
  * 'a' is an event of t_1, i.e. $t_1 = a\,t_1'$ and $t \in t_1' \parallel t_2$ $\quad (i3)$

    We know that, $a\,t_1' \in [\![h]\!][\![\Delta]\!]\rho$ by $i2$ and Lemma 26

    $\Longrightarrow \quad \Delta, \Gamma \vdash\ h \xrightarrow{a} h'$ and $t_1' \in [\![h']\!][\![\Delta]\!]\rho$ by *IH* $\quad (i4)$

    $\overset{\text{ASYM-L}}{\Longrightarrow} \Delta, \Gamma \vdash\ h$ **where** $x :\in g \xrightarrow{a} h'$ **where** $x :\in g$

    By $i1$, $\exists u \in Val.\ a\,t_1' \in [\![h]\!][\![\Delta]\!]\rho[x = u]$

    $\Longrightarrow \quad \Delta, \Gamma[x = u] \vdash\ h \xrightarrow{a} h'$ and $t_1' \in [\![h']\!][\![\Delta]\!]\rho[x = u]$ by *IH*

    $\Longrightarrow \quad t_1' \in \bigcup_{v \in Val} [\![h']\!][\![\Delta]\!]\rho[x = v]$

    $\Longrightarrow \quad t \in [\![h'$ **where** $x :\in g]\!][\![\Delta]\!]\rho$ by $i1, i3$
  * 'a' is an event of t_2, i.e. $t_2 = a\,t_2'$ and $t \in t_1 \parallel t_2'$ $\quad (i5)$

    $\Delta, \Gamma \vdash\ g \xrightarrow{a} g'$ and $t_2' \in [\![g']\!][\![\Delta]\!]\rho$ by *IH* for g $\quad (i6)$

    $\overset{\text{ASYM-R}}{\Longrightarrow} \Delta, \Gamma \vdash\ h$ **where** $x :\in g \xrightarrow{a} h$ **where** $x :\in g'$

    Also, $t \in t_1 <_x t_2'$ by $i2, i5$

    $\Longrightarrow \quad t \in [\![h$ **where** $x :\in g']\!][\![\Delta]\!]\rho$ by $i1, i6$
- $\bar{R}(x, t_1),\ t_2 = t_{21}!w\,t_{22},\ \bar{P}(t_{21})$

  $\Longrightarrow a\,t \in t_1 \parallel t_{21}\,\tau$ $\quad (i7)$
  * 'a' is an event of t_1, i.e. $t_1 = a\,t_1'$ and $t \in t_1' \parallel t_{21}\,\tau$

    . . . It's exactly the same as the previous case for t_1
  * 'a' is an event of t_{21}, i.e. $t_{21} = a\,t_{21}'$ and $t \in t_1 \parallel t_{21}'\,\tau$

    . . . It's exactly the same as the previous case for t_2
  * t_{21} is empty, $a = \tau$ and $t = t_1$

    $\Delta, \Gamma \vdash\ g \xrightarrow{!w} g'$ and $t_{22} \in [\![g']\!][\![\Delta]\!]\rho$ by *IH* for g

    $\overset{\text{ASYM-P}}{\Longrightarrow} \Delta, \Gamma \vdash\ h$ **where** $x :\in g \xrightarrow{\tau} [w/x]h$

    Suffices to show that $t_1 \in [\![[w/x]h]\!][\![\Delta]\!]\rho$

    We know that $t_1 \in [\![h]\!][\![\Delta]\!]\rho_{-x}$ by $i7$ and Lemma 26

    $\Longrightarrow \quad t_1 \in [\![h]\!][\![\Delta]\!]\rho[x = w]$ by Corollary 6

    But $\bar{R}(x, t_1)$ so $\quad t_1 \in [\![[w/x]h]\!][\![\Delta]\!]\rho$ by Lemma 30
- $t_1 = t_{11}[w/x]\,t_{12}, \bar{R}(x, t_{11}),\ t_2 = t_{21}!w\,t_{22},\ \bar{P}(t_{21})$

  $\Longrightarrow \quad a\,t \in (t_{11} \parallel t_{21}\tau)(t_{12}\backslash[w/x])$
  * 'a' is an event of t_{11},

    i.e. $t_{11} = a\,t_{11}'$ and $t \in (t_{11}' \parallel t_{21}\,\tau)(t_{12}\backslash[w/x])$

    $\Longrightarrow \quad t \in (t_{11}'[w/x]t_{12} <_x t_2)$ $\quad (i8)$

By Lemma 25, $t_1 \in [\![h]\!][\![\Delta]\!]\rho[x = w]$
$\Longrightarrow \quad \Delta, \Gamma[x = w] \vdash \; h \xrightarrow{a} h'$ and
$\quad t'_{11}[w/x]t_{12} \in [\![h']\!][\![\Delta]\!]\rho[x = w]$ by *IH* $(i9)$
$\Longrightarrow \quad \Delta, \Gamma \vdash \; h \xrightarrow{a} h'$ by Lemma 32
$\stackrel{\text{ASYM-L}}{\Longrightarrow} \Delta, \Gamma \vdash \; h \,\textbf{where}\, x :\in g \xrightarrow{a} h' \,\textbf{where}\, x :\in g$
Also, by $i8$ and $i9$ $t \in [\![h' \,\textbf{where}\, x :\in g]\!][\![\Delta]\!]\rho$

- ‘a’ is an event of t_{21},
  i.e. $t_{21} = a\,t'_{21}$ and $t \in (t_{11} \parallel t'_{21}\,\tau)(t_{12}\backslash[w/x])$
  $\Longrightarrow \quad t \in (t_1 <_x t'_{21}!w\,t_{22})$ $(i10)$
  $\Delta, \Gamma \vdash \; g \xrightarrow{a} g'$ and $t'_{21}!w\,t_{22} \in [\![g']\!][\![\Delta]\!]\rho$ by *IH* $(i11)$
  $\stackrel{\text{ASYM-R}}{\Longrightarrow} \Delta, \Gamma \vdash \; h \,\textbf{where}\, x :\in g \xrightarrow{a} h \,\textbf{where}\, x :\in g'$
  and $t \in [\![h \,\textbf{where}\, x :\in g']\!][\![\Delta]\!]\rho$ by $i10, i11$
- t_{21} is empty, $a = \tau$ and $t = t_{11}(t_{21}\backslash[w/x]) = t_1\backslash[w/x]$ $(i12)$
  $\Delta, \Gamma \vdash \; g \xrightarrow{!w} g'$ and $t_{22} \in [\![g']\!][\![\Delta]\!]\rho$ by *IH*
  $\stackrel{\text{ASYM-P}}{\Longrightarrow} \Delta, \Gamma \vdash \; h \,\textbf{where}\, x :\in g \xrightarrow{\tau} [w/x]h$
  By Lemma 25, $t_1 \in [\![h]\!][\![\Delta]\!]\rho[x = w]$
  $\Longrightarrow \quad t_1\backslash[w/x] \in ([\![h]\!][\![\Delta]\!]\rho[x = w])\backslash[w/x]$
  $\Longrightarrow \quad t \in [\![[w/x]h]\!][\![\Delta]\!]\rho$ by $i12$ and Lemma 30

□

Lemma 38. *If* $t_1\,t_2 \in [\![f]\!][\![\Delta]\!]\rho$ *then* $\Delta, \Gamma \vdash \; f \xrightarrow{t_1}{}^{*} f'$ *and* $t_2 \in [\![f']\!][\![\Delta]\!]\rho$ *where* $\rho = \rho_0[x_1 = v_1]\ldots[x_m = v_m]$ *and* $\Gamma = \{(x_1, v_1), \ldots, (x_m, v_m)\}$

Proof. By induction on $|t_1|$ □

Theorem 8 (Adequacy). *If* $\Gamma = \{(x_1, v_1), \ldots, (x_m, v_m)\}$, $\rho = \rho_0[x_1 = v_1]\ldots[x_m = v_m]$, *x's are pairwise distinct, then*

$$t \in [\![f]\!][\![\Delta]\!]\rho \quad \textit{implies} \quad \Delta, \Gamma \vdash \; f \xrightarrow{t}{}^{*} f'$$

Proof. By induction on $|t|$

- If $|t| = 0 \Leftrightarrow t = \varepsilon$, then f reduces to itself in 0 steps.
- If $t = a\,t'$ then $a\,t' \in [\![f]\!][\![\Delta]\!]\rho$
  $\Longrightarrow \quad \Delta, \Gamma \vdash \; f \xrightarrow{a} f'$ and $t' \in [\![f']\!][\![\Delta]\!]\rho$ by Lemma 37
  $\Longrightarrow \quad \Delta, \Gamma \vdash \; f' \xrightarrow{t'}{}^{*} f''$ by *IH* for t'
  $\Longrightarrow \quad \Delta, \Gamma \vdash \; f \xrightarrow{a} f' \xrightarrow{t'}{}^{*} f''$
  $\Longrightarrow \quad \Delta, \Gamma \vdash \; f \xrightarrow{t}{}^{*} f''$ □

G Strong Bisimulation

To improve readability, in this section we use the notation $\langle a, b\rangle$ for ordered pairs instead of (a, b).

Definition 22 (Strong Bisimulation). *The binary relation* $\mathfrak{R}$ *on processes is a* Δ*-bisimulation iff*

1. $\mathfrak{R}$ *is symmetric*
2. *for any* $\langle f, g\rangle \in \mathfrak{R}$ *and for any* Γ *if* $\Delta, \Gamma \vdash f \xrightarrow{a} f'$ *then* $\Delta, \Gamma \vdash g \xrightarrow{a} g'$ *and* $\langle f', g'\rangle \in \mathfrak{R}$

Definition 23 (Largest Strong-Bisimulation). $\sim_\Delta \triangleq \bigcup\{\, \mathfrak{R} \mid \mathfrak{R}$ *is a* Δ*-bisim.*$\}$

Definition 24 (Strong-Bisimulation up to $\sim_\Delta$). *The binary relation* $\mathfrak{R}$ *on processes is a* Δ*-bisimulation up to* $\sim_\Delta$*, if* $\sim_\Delta \mathfrak{R} \sim_\Delta$ *is a* Δ*-bisimulation*

Lemma 39. $\sim_\Delta$ *is an equivalence relation*

Lemma 40. $f \mid 0 \sim_\Delta f$

Lemma 41. $f \mid g \sim_\Delta g \mid f$

Lemma 42. $f \mid (g \mid h) \sim_\Delta (f \mid g) \mid h$

Proof. $\mathfrak{R}_1 = \{\langle f \mid (g \mid h), (f \mid g) \mid h\rangle\}$
$\mathfrak{R} = \mathfrak{R}_1 \cup {\mathfrak{R}_1}^{-1}$ is a Δ-bisimulation
If f takes a step, $\Delta, \Gamma \vdash f \xrightarrow{a} f'$ then
$\Longrightarrow \quad \Delta, \Gamma \vdash f \mid (g \mid h) \xrightarrow{a} f' \mid (g \mid h) \equiv f' \mid (g \mid h)$
Also,
$\Longrightarrow \quad \Delta, \Gamma \vdash f \mid g \xrightarrow{a} f' \mid g$
$\Longrightarrow \quad \Delta, \Gamma \vdash (f \mid g) \mid h \xrightarrow{a} (f' \mid g) \mid h$
But $\langle f' \mid (g \mid h), (f' \mid g) \mid h\rangle \in \mathfrak{R}$. Similarly if g or h takes a step □

Lemma 43. $[v/x]f \sim_\Delta [v/x]g$ *when* $f \sim_\Delta g$

Proof. $\mathfrak{R} = \{\, \langle [v/x]f, [v/x]g\rangle \mid f \sim_\Delta g\}$ is a Δ-bisimulation.
$\mathfrak{R}$ is clearly symmetric. When $[v/x]f$ takes a step, $\Delta, \Gamma \vdash [v/x]f \xrightarrow{a} f'$
we must show that for some g' $\Delta, \Gamma \vdash [v/x]g \xrightarrow{a} g'$ and $f' \,\mathfrak{R}\, g'$
By the contrapositive of Lemma 31 we know that $\bar{R}(x, a)$ because $x \notin \text{fv}([v/x]f)$
Then, by Lemma 34 there are two cases for the steps of f.

a) $\Delta, \Gamma \vdash f \xrightarrow{a} f''$ and $[v/x]f'' \equiv f'$
But $f \sim_\Delta g$ so $\Delta, \Gamma \vdash g \xrightarrow{a} g''$ and $f'' \sim_\Delta g''$ $\quad (I)$
By Lemma 32 we get, $\Delta, \Gamma \vdash [v/x]g \xrightarrow{a} [v/x]g''$
$\Longrightarrow \quad f' \equiv [v/x]f'' \,\mathfrak{R}\, [v/x]g'' \equiv g'$ by I

b) $\Delta, \Gamma[x = v] \vdash\ f \overset{[v/x]}{\rightarrow}\ f_1 \overset{a}{\rightarrow}\ f_2 \qquad [v/x]f \equiv [v/x]f_1,\ f' \equiv [v/x]f_2$
But $f \sim_\Delta g$ so
$\Delta, \Gamma[x = v] \vdash\ g \overset{[v/x]}{\rightarrow}\ g_1 \qquad f_1 \sim_\Delta g_1, \quad [v/x]g \equiv [v/x]g_1$ by Lem. 31 (II)
$\Longrightarrow \quad \Delta, \Gamma[x = v] \vdash\ g_1 \overset{a}{\rightarrow}\ g_2 \qquad f_2 \sim_\Delta g_2$ (III)
$\Longrightarrow \quad \Delta, \Gamma \vdash\ g_1 \overset{a}{\rightarrow}\ g_2$ by Lemma 32.2
$\Longrightarrow \quad \Delta, \Gamma \vdash\ [v/x]g_1 \overset{a}{\rightarrow}\ [v/x]g_2$ by Lemma 32.1
$\Longrightarrow \quad \Delta, \Gamma \vdash\ [v/x]g \overset{a}{\rightarrow}\ [v/x]g_2$ by II
$\Longrightarrow \quad f' \equiv [v/x]f_2 \,\mathfrak{R}\, [v/x]g_2 \equiv g'$ by III □

Lemma 44. $f \sim_\Delta (f \textbf{ where } x :\in \mathbf{0})$ *when* $x \notin \mathrm{fv}(f)$

Lemma 45. $\sim_\Delta$ *is a congruence relation*

Proof. We show appropriate bisimulations for all possible contexts:

a) $\mathfrak{R} = \{\, \langle f \mid h, g \mid h \rangle \mid f \sim_\Delta g \}$ is a Δ-bisimulation.
If h takes a step, trivial.
If f takes a step, $\Delta, \Gamma \vdash\ f \overset{a}{\rightarrow}\ f'$
$\Longrightarrow \quad \Delta, \Gamma \vdash\ f \mid h \overset{a}{\rightarrow}\ f' \mid h$
But $f \sim_\Delta g$ so, $\Delta, \Gamma \vdash\ g \overset{a}{\rightarrow}\ g'$ and $f' \sim_\Delta g'$
$\Longrightarrow \quad \Delta, \Gamma \vdash\ g \mid h \overset{a}{\rightarrow}\ g' \mid h$
and $\langle f' \mid h, g' \mid h \rangle \in \mathfrak{R}$
Similarly for the transitions of g.

b) $\mathfrak{R} = \{\, \langle h \mid f, h \mid g \rangle \mid f \sim_\Delta g \}$ is a Δ-bisimulation.
As above.

c) $\mathfrak{R} = \{\, \langle (f > x > h) \mid d, (g > x > h) \mid d \rangle \mid f \sim_\Delta g \}$
$\mathfrak{R}$ is a Δ-bisimulation up to $\sim_\Delta$.
The only interesting case is when f publishes, $\Delta, \Gamma \vdash\ f \overset{!v}{\rightarrow}\ f'$
$\Longrightarrow \quad \Delta, \Gamma \vdash\ f > x > h \overset{\tau}{\rightarrow}\ (f' > x > h) \mid [v/x]h$
$\Longrightarrow \quad \Delta, \Gamma \vdash\ (f > x > h) \mid d \overset{\tau}{\rightarrow}\ ((f' > x > h) \mid [v/x]h) \mid d$
But $f \sim_\Delta g$ so, $\Delta, \Gamma \vdash\ g \overset{!v}{\rightarrow}\ g'$ and $f' \sim_\Delta g'$
$\Longrightarrow \quad \Delta, \Gamma \vdash\ g > x > h \overset{\tau}{\rightarrow}\ (g' > x > h) \mid [v/x]h$
$\Longrightarrow \quad \Delta, \Gamma \vdash\ (g > x > h) \mid d \overset{\tau}{\rightarrow}\ ((g' > x > h) \mid [v/x]h) \mid d$
Then,
$((f' > x > h) \mid [v/x]h) \mid d \quad \sim_\Delta$ by Lemma 42
$(f' > x > h) \mid ([v/x]h \mid d) \quad \mathfrak{R}$
$(g' > x > h) \mid ([v/x]h \mid d) \quad \sim_\Delta$ by Lemma 42
$((g' > x > h) \mid [v/x]h) \mid d$
Similarly for g's transitions.
The desired result follows by Lemma 40 when $d \equiv \mathbf{0}$.

d) $\mathfrak{R} = \{\, \langle (h > x > f) \mid d_1, (h > x > g) \mid d_2 \rangle \mid f \sim_\Delta g,\ d_1 \sim_\Delta d_2 \}$
$\mathfrak{R}$ is a Δ-bisimulation up to $\sim_\Delta$.
The only interesting case is when h publishes, $\Delta, \Gamma \vdash\ h \overset{!v}{\rightarrow}\ h'$
$\Longrightarrow \quad \Delta, \Gamma \vdash\ h > x > f \overset{\tau}{\rightarrow}\ (h' > x > f) \mid [v/x]f$
$\Longrightarrow \quad \Delta, \Gamma \vdash\ (h > x > f) \mid d_1 \overset{\tau}{\rightarrow}\ ((h' > x > f) \mid [v/x]f) \mid d_1$

Also, $\Delta, \Gamma \vdash\ h >x> g \xrightarrow{\tau} (h' >x> g) \mid [v/x]g$
$\implies \quad \Delta, \Gamma \vdash\ (h >x> g) \mid d_2 \xrightarrow{\tau} ((h' >x> g) \mid [v/x]g) \mid d_2$
By Lemma 43, $\quad [v/x]f \sim_\Delta [v/x]g$
Then,
$((h' >x> f) \mid [v/x]f) \mid d_1 \quad \sim_\Delta$ by case b
$((h' >x> f) \mid [v/x]f) \mid d_2 \quad \sim_\Delta$ by Lemma 42
$(h' >x> f) \mid ([v/x]f \mid d_2) \quad \sim_\Delta$ by cases a,b
$(h' >x> f) \mid ([v/x]g \mid d_2) \quad \mathfrak{R}$
$(h' >x> g) \mid ([v/x]g \mid d_2) \quad \sim_\Delta$ by Lemma 42
$((h' >x> g) \mid [v/x]g) \mid d_2$
The desired result follows by Lemma 40 when $d_1 \equiv \mathbf{0}, d_2 \equiv \mathbf{0}$.

e) $\mathfrak{R} = \{\, \langle f \textbf{ where } x :\in h, g \textbf{ where } x :\in h \rangle \mid f \sim_\Delta g\}$
$\mathfrak{R}$ is a Δ-bisimulation up to $\sim_\Delta$.
The only interesting case is when h publishes, $\quad \Delta, \Gamma \vdash\ h \xrightarrow{!v} h'$
$\implies \quad \Delta, \Gamma \vdash\ f \textbf{ where } x :\in h \xrightarrow{\tau} [v/x]f$
Also, $\Delta, \Gamma \vdash\ g \textbf{ where } x :\in h \xrightarrow{\tau} [v/x]g$
$[v/x]f \quad \sim_\Delta$ by Lemma 44
$[v/x]f \textbf{ where } x :\in \mathbf{0} \quad \mathfrak{R}$ by Lemma 43
$[v/x]g \textbf{ where } x :\in \mathbf{0} \quad \sim_\Delta$ by Lemma 44
$[v/x]g$

f) $\mathfrak{R}_1 = \{\, \langle h \textbf{ where } x :\in f, h \textbf{ where } x :\in g \rangle \mid f \sim_\Delta g\}$
$\mathfrak{R} = \mathfrak{R}_1 \cup \mathcal{ID}$ is a Δ-bisimulation.
If f publishes, $\quad \Delta, \Gamma \vdash\ f \xrightarrow{!v} f'$
$\implies \quad \Delta, \Gamma \vdash\ h \textbf{ where } x :\in f \xrightarrow{\tau} [v/x]h$
But $f \sim_\Delta g$ so, $\Delta, \Gamma \vdash\ g \xrightarrow{!v} g'$ and $f' \sim_\Delta g'$
$\implies \quad \Delta, \Gamma \vdash\ h \textbf{ where } x :\in g \xrightarrow{\tau} [v/x]h$
and $\quad \langle [v/x]h, [v/x]h \rangle \in \mathfrak{R}$
If f takes a non-publication step, $\quad \Delta, \Gamma \vdash\ f \xrightarrow{a} f'$
$\implies \quad \Delta, \Gamma \vdash\ h \textbf{ where } x :\in f \xrightarrow{a} h \textbf{ where } x :\in f'$
But $f \sim_\Delta g$ so, $\Delta, \Gamma \vdash\ g \xrightarrow{a} g'$ and $f' \sim_\Delta g'$
$\implies \quad \Delta, \Gamma \vdash\ h \textbf{ where } x :\in g \xrightarrow{a} h \textbf{ where } x :\in g'$
and $\quad \langle h \textbf{ where } x :\in f', h \textbf{ where } x :\in g' \rangle \in \mathfrak{R}$ □

Lemma 46. $(f \mid g) >x> h \sim_\Delta (f >x> h) \mid (g >x> h)$

Proof. $\mathfrak{R}_1 = \{\langle ((f \mid g) >x> h) \mid d, ((f >x> h) \mid (g >x> h)) \mid d \rangle\}$
$\mathfrak{R} = \mathfrak{R}_1 \cup {\mathfrak{R}_1}^{-1}$ is a Δ-bisimulation up to $\sim_\Delta$
The only interesting case is when f publishes, $\quad \Delta, \Gamma \vdash\ f \xrightarrow{!v} f'$
$\implies \quad \Delta, \Gamma \vdash\ f \mid g \xrightarrow{!v} f' \mid g$
$\implies \quad \Delta, \Gamma \vdash\ (f \mid g) >x> h \xrightarrow{\tau} ((f' \mid g) >x> h) \mid [v/x]h$
$\implies \quad \Delta, \Gamma \vdash\ ((f \mid g) >x> h) \mid d \xrightarrow{\tau} (((f' \mid g) >x> h) \mid [v/x]h) \mid d$
Also,
$\implies \quad \Delta, \Gamma \vdash\ f >x> h \xrightarrow{\tau} (f' >x> h) \mid [v/x]h$
$\implies \quad \Delta, \Gamma \vdash\ (f >x> h) \mid (g >x> h) \xrightarrow{\tau} ((f' >x> h) \mid [v/x]h) \mid (g >x> h)$

$\Longrightarrow \quad \Delta, \Gamma \vdash ((f >x> h) \mid (g >x> h)) \mid d \xrightarrow{\tau}$
$(((f' >x> h) \mid [v/x]h) \mid (g >x> h)) \mid d$

But then,

$(((f' >x> h) \mid [v/x]h) \mid (g >x> h)) \mid d \ \sim_\Delta$ by Lemmas 41,42

$((f' >x> h) \mid (g >x> h)) \mid ([v/x]h \mid d) \quad \mathfrak{R}$

$((f' \mid g) >x> h) \mid ([v/x]h \mid d)$

The desired result follows by Lemma 40 when $d \equiv \mathbf{0}$. □

Lemma 47. $f >x> (g >y> h) \sim_\Delta (f >x> g) >y> h \quad$ *if* $x \notin \text{fv}(h)$

Proof. $\mathfrak{R}_1 = \{\langle (f >x> (g >y> h)) \mid d, ((f >x> g) >y> h) \mid d \rangle\}$

$\mathfrak{R} = \mathfrak{R}_1 \cup {\mathfrak{R}_1}^{-1}$ is a Δ-bisimulation up to $\sim_\Delta$ if $x \notin \text{fv}(h)$

The only interesting case is when f publishes, $\quad \Delta, \Gamma \vdash f \xrightarrow{!v} f'$

$\Longrightarrow \quad \Delta, \Gamma \vdash f >x> (g >y> h) \xrightarrow{\tau}$
$(f' >x> (g >y> h)) \mid [v/x](g >y> h)$

$\Longrightarrow \quad \Delta, \Gamma \vdash ((f >x> g) >y> h) \mid d \xrightarrow{\tau}$
$((f' >x> (g >y> h)) \mid [v/x](g >y> h)) \mid d$

Also,

$\Longrightarrow \quad \Delta, \Gamma \vdash f >x> g \xrightarrow{\tau} (f' >x> g) \mid [v/x]g$

$\Longrightarrow \quad \Delta, \Gamma \vdash (f >x> g) >y> h \xrightarrow{\tau} ((f' >x> g) \mid [v/x]g) >y> h$

$\Longrightarrow \quad \Delta, \Gamma \vdash ((f >x> g) >y> h) \mid d \xrightarrow{\tau}$
$(((f' >x> g) \mid [v/x]g) >y> h) \mid d$

By Lemma 46,

$(((f' >x> g) \mid [v/x]g) >y> h) \mid d \quad \sim_\Delta$

$(((f' >x> g) >y> h) \mid ([v/x]g >y> h)) \mid d \quad \mathfrak{R}$

$((f' >x> (g >y> h)) \mid [v/x](g >y> h)) \mid d$ □

Lemma 48. $(f \mid g)$ **where** $x :\in h \sim_\Delta (f$ **where** $x :\in h) \mid g \quad$ *if* $x \notin \text{fv}(g)$

Proof. $\mathfrak{R}_1 = \{\langle (f \mid g)$ **where** $x :\in h, (f$ **where** $x :\in h) \mid g \rangle\}$

$\mathfrak{R} = \mathfrak{R}_1 \cup {\mathfrak{R}_1}^{-1} \cup \mathcal{ID}$ is a Δ-bisimulation if $x \notin \text{fv}(g)$

Let $\quad \Delta, \Gamma \vdash g \xrightarrow{a} g'$

We know that $x \notin \text{fv}(g)$ so, by Lemma 31, 'a' is not a receive for x. Then,

$\Longrightarrow \quad \Delta, \Gamma \vdash f \mid g \xrightarrow{a} f \mid g'$

$\Longrightarrow \quad \Delta, \Gamma \vdash (f \mid g)$ **where** $x :\in h \xrightarrow{a} (f \mid g')$ **where** $x :\in h$

Also, $\Delta, \Gamma \vdash g \xrightarrow{a} g'$

$\Longrightarrow \quad \Delta, \Gamma \vdash (f$ **where** $x :\in h) \mid g \xrightarrow{a} (f$ **where** $x :\in h) \mid g'$

But by Lemma 33, $x \notin \text{fv}(g')$

$\Longrightarrow \quad ((f \mid g')$ **where** $x :\in h) \quad \mathfrak{R} \quad ((f$ **where** $x :\in h) \mid g')$

The only interesting case left is when h publishes, $\quad \Delta, \Gamma \vdash h \xrightarrow{!v} h'$

$\Longrightarrow \quad \Delta, \Gamma \vdash (f \mid g)$ **where** $x :\in h \xrightarrow{\tau} [v/x](f \mid g)$

$\equiv [v/x]f \mid g$ because $x \notin \text{fv}(g)$

Also,

$\Longrightarrow \quad \Delta, \Gamma \vdash f$ **where** $x :\in h \xrightarrow{\tau} [v/x]f$

$\Longrightarrow \quad \Delta, \Gamma \vdash (f$ **where** $x :\in h) \mid g \xrightarrow{\tau} [v/x]f \mid g$

And obviously, $\langle [v/x]f \mid g, [v/x]f \mid g \rangle \in \mathcal{ID}$ □

Lemma 49. $(f > y > g)$ **where** $x :\in h \sim_\Delta (f$ **where** $x :\in h) > y > g$
if $x \notin \mathrm{fv}(g)$

Proof.
$\mathfrak{R}_1 = \{\langle((f > y > g) \textbf{ where } x :\in h) \mid d, ((f \textbf{ where } x :\in h) > y > g) \mid d\rangle\}$
$\mathfrak{R} = \mathfrak{R}_1 \cup {\mathfrak{R}_1}^{-1} \cup \mathcal{ID}$ is a Δ-bisimulation up to $\sim_\Delta$ if $x \notin \mathrm{fv}(g)$
We look only at the publication steps of h and f.
$\Delta, \Gamma \vdash\ h \xrightarrow{!v} h'$
$\Longrightarrow\quad \Delta, \Gamma \vdash\ (f > y > g) \textbf{ where } x :\in h \xrightarrow{\tau} [v/x](f > y > g)$
$\Longrightarrow\quad \Delta, \Gamma \vdash\ ((f > y > g) \textbf{ where } x :\in h) \mid d \xrightarrow{\tau} [v/x](f > y > g) \mid d$
Also,
$\Longrightarrow\quad \Delta, \Gamma \vdash\ f \textbf{ where } x :\in h \xrightarrow{\tau} [v/x]f$
$\Longrightarrow\quad \Delta, \Gamma \vdash\ (f \textbf{ where } x :\in h) > y > g \xrightarrow{\tau} [v/x]f > y > g$
$\Longrightarrow\quad \Delta, \Gamma \vdash\ ((f \textbf{ where } x :\in h) > y > g) \mid d \xrightarrow{\tau} ([v/x]f > y > g) \mid d$
But $x \notin \mathrm{fv}(g)$ so
$\langle [v/x](f > y > g) \mid d, ([v/x]f > y > g) \mid d\rangle \in \mathcal{ID}$
If f publishes, $\Delta, \Gamma \vdash\ f \xrightarrow{!v} f'$ then,
$\Longrightarrow\quad \Delta, \Gamma \vdash\ f > y > g \xrightarrow{\tau} (f' > y > g) \mid [v/y]g$
$\Longrightarrow\quad \Delta, \Gamma \vdash\ (f > y > g) \textbf{ where } x :\in h \xrightarrow{\tau} ((f' > y > g) \mid [v/y]g) \textbf{ where } x :\in h$
$\Longrightarrow\quad \Delta, \Gamma \vdash\ ((f > y > g) \textbf{ where } x :\in h) \mid d \xrightarrow{\tau}$
$\qquad (((f' > y > g) \mid [v/y]g) \textbf{ where } x :\in h) \mid d$
Also,
$\Delta, \Gamma \vdash\ f \textbf{ where } x :\in h \xrightarrow{!v} f' \textbf{ where } x :\in h$
$\Delta, \Gamma \vdash\ (f \textbf{ where } x :\in h) > y > g \xrightarrow{\tau} ((f' \textbf{ where } x :\in h) > y > g) \mid [v/y]g$
$\Delta, \Gamma \vdash\ ((f \textbf{ where } x :\in h) > y > g) \mid d \xrightarrow{\tau}$
$\qquad (((f' \textbf{ where } x :\in h) > y > g) \mid [v/y]g) \mid d$
But $x \notin \mathrm{fv}(g)$ so by Lemma 48
$((f' > y > g) \mid [v/y]g) \textbf{ where } x :\in h \quad \sim_\Delta$
$((f' > y > g) \textbf{ where } x :\in h) \mid [v/y]g$
which by Lemma 45 yields
$(((f' > y > g) \mid [v/y]g) \textbf{ where } x :\in h) \mid d \quad \sim_\Delta$
$(((f' > y > g) \textbf{ where } x :\in h) \mid [v/y]g) \mid d$
By Lemma 42, the last process is strongly bisimilar to
$((f' > y > g) \textbf{ where } x :\in h) \mid ([v/y]g \mid d) \quad \mathfrak{R}$
$((f' \textbf{ where } x :\in h) > y > g) \mid ([v/y]g \mid d) \quad \sim_\Delta$
$(((f' \textbf{ where } x :\in h) > y > g) \mid [v/y]g) \mid d$
The desired result follows by Lemma 40 when $d \equiv \mathbf{0}$. □

Lemma 50. $(f$ **where** $x :\in g)$ **where** $y :\in h \sim_\Delta (f$ **where** $y :\in h)$ **where** $x :\in g$
if $x \notin \mathrm{fv}(h), y \notin \mathrm{fv}(g)$

Proof. The proof is similar to the previous proofs □

Lemma 51. *If* $\Gamma = \{\langle x_1, v_1\rangle, \ldots, \langle x_m, v_m\rangle\}$ *then*

$$f \sim_\Delta g \Rightarrow (\Delta, \Gamma \vdash\ f \xrightarrow{t}{}^{*} f' \Leftrightarrow \Delta, \Gamma \vdash\ g \xrightarrow{t}{}^{*} g')$$

Proof. By induction on $|t|$. Note that since Γ is a partial function, and not an arbitrary relation, we can assume that x_i's are pairwise distinct. $\Box$

Theorem 9. *If $f \sim_\Delta g$ then for any ρ it holds that $[\![f]\!][\![\Delta]\!]\rho = [\![g]\!][\![\Delta]\!]\rho$*

Proof. Let $t \in [\![f]\!][\![\Delta]\!]\rho$

$\implies \quad \Delta, \Gamma \vdash f \xrightarrow{t}{}^{*} f'$ by Theorem 8

$\implies \quad \Delta, \Gamma \vdash g \xrightarrow{t}{}^{*} g'$ by Lemma 51

$\implies \quad t \in [\![g]\!][\![\Delta]\!]\rho$ by Theorem 7

$\implies \quad [\![f]\!][\![\Delta]\!]\rho \subseteq [\![g]\!][\![\Delta]\!]\rho$

In the same way we get $[\![g]\!][\![\Delta]\!]\rho \subseteq [\![f]\!][\![\Delta]\!]\rho$ $\Box$

The next lemma may seem counterintuitive on a first reading. It is the main lemma in order to show that the events following a publication are not caused by it; they could have preceded it.

Lemma 52. *If $\Delta, \Gamma \vdash f \xrightarrow{!v} f'$ then $f \sim_\Delta let(v) \mid f'$*

Proof. By induction on the height of the derivation

– $$\frac{}{\Delta, \Gamma \vdash let(v) \xrightarrow{!v} \mathbf{0}}$$

Clearly, $let(v) \sim_\Delta let(v) \mid \mathbf{0}$

– $$\frac{\Delta, \Gamma \vdash h \xrightarrow{!v} h'}{\Delta, \Gamma \vdash h \mid g \xrightarrow{!v} h' \mid g}$$

By *IH*, $h \sim_\Delta let(v) \mid h'$

$\implies \quad h \mid g \sim_\Delta (let(v) \mid h') \mid g$ by Lemma 45

$\implies \quad h \mid g \sim_\Delta let(v) \mid (h' \mid g)$ by Lemma 42

Similarly for SYM-R

– $$\frac{\Delta, \Gamma \vdash h \xrightarrow{!v} h'}{\Delta, \Gamma \vdash h \textbf{ where } x :\in g \xrightarrow{!v} h' \textbf{ where } x :\in g}$$

By *IH*, $h \sim_\Delta let(v) \mid h'$

$\implies \quad h \textbf{ where } x :\in g \sim_\Delta (let(v) \mid h') \textbf{ where } x :\in g$ by Lemma 45

$\implies \quad h \textbf{ where } x :\in g \sim_\Delta (h' \textbf{ where } x :\in g) \mid let(v)$ by Lemma 48

$\implies \quad h \textbf{ where } x :\in g \sim_\Delta let(v) \mid (h' \textbf{ where } x :\in g)$ by Lemma 41

These are the only rules where f can publish $\Box$

H Various properties of the trace-semantics

Lemma 53. *If* $s = !v\, s'$ *and* $s \in [\![f]\!][\![\Delta]\!]\rho$ *then* $(!v \parallel s') \subseteq [\![f]\!][\![\Delta]\!]\rho$

Proof. By Lemma 37, $\Delta, \Gamma \vdash f \xrightarrow{!v} f'$ and $s' \in [\![f']\!][\![\Delta]\!]\rho$

$\Longrightarrow \quad f \sim_\Delta let(v) \mid f'$ by Lemma 52

$\Longrightarrow \quad [\![f]\!][\![\Delta]\!]\rho = [\![let(v) \mid f']\!][\![\Delta]\!]\rho$ by Theorem 9

$\Longrightarrow \quad [\![let(v)]\!][\![\Delta]\!]\rho \parallel [\![f']\!][\![\Delta]\!]\rho \subseteq [\![f]\!][\![\Delta]\!]\rho$

$\Longrightarrow \quad (!v \parallel s') \subseteq [\![f]\!][\![\Delta]\!]\rho$ by monotonicity of $\parallel$ □

Lemma 54. *If* $s = s_1 !v\, s_2$ *and* $\bar{P}(s_1)$ *and* $s \in [\![f]\!][\![\Delta]\!]\rho$
then $s_1(!v \parallel s_2) \subseteq [\![f]\!][\![\Delta]\!]\rho$

Proof. By induction on $|s_1|$

- If $|s_1| = 0$ immediate by Lemma 53
- If $s_1 = a\, s_1'$
  By Lemma 37 we get $\Delta, \Gamma \vdash f \xrightarrow{a} f'$ and $s_1' !v\, s_2 \in [\![f']\!][\![\Delta]\!]\rho$
  By *IH* $s_1' (!v \parallel s_2) \subseteq [\![f']\!][\![\Delta]\!]\rho$ (I)
  Let $t \in s_1(!v \parallel s_2)$
  $\Longrightarrow \quad t \in a\, s_1'(!v \parallel s_2)$
  $\Longrightarrow \quad t \equiv a\, t' \quad \wedge \quad t' \in s_1'(!v \parallel s_2)$
  $\Longrightarrow \quad t \equiv a\, t' \quad \wedge \quad t' \in [\![f']\!][\![\Delta]\!]\rho$ by I
  $\Longrightarrow \quad a\, t' \in [\![f]\!][\![\Delta]\!]\rho$ by Lemma 35
  $\Longrightarrow \quad t \in [\![f]\!][\![\Delta]\!]\rho$
  $\Longrightarrow \quad s_1(!v \parallel s_2) \subseteq [\![f]\!][\![\Delta]\!]\rho$ □

We showed the non-causality of the first publication in a trace, but this can be generalized to any publication.

Lemma 55. *If* $s = s_1 !v\, s_2$ *and* $s \in [\![f]\!][\![\Delta]\!]\rho$ *then* $s_1(!v \parallel s_2) \subseteq [\![f]\!][\![\Delta]\!]\rho$

Proof. By induction on the number of publications in s

- If s has one publication then we get a special case of Lemma 54
- If s has more than one publication then it is of the form $s_1 !v_1\, s_2 !v_2\, s_3$ where $!v_1$ is the first publication and $!v_2$ isn't necessarily the second publication.
  By Lemma 54 we know $s_1(!v_1 \parallel s_2 !v_2\, s_3) \subseteq [\![f]\!][\![\Delta]\!]\rho$
  so it suffices to show that $s_1 !v_1\, s_2(!v_2 \parallel s_3) \subseteq [\![f]\!][\![\Delta]\!]\rho$
  By Lemma 38 we get
  $\Delta, \Gamma \vdash f \xrightarrow{s_1 !v_1}{}^{*} f'$ and $s_2 !v_2\, s_3 \in [\![f']\!][\![\Delta]\!]\rho$
  $\Longrightarrow \quad s_2(!v_2 \parallel s_3) \subseteq [\![f']\!][\![\Delta]\!]\rho$ by *IH* (I)
  Let $t \in s_1 !v_1\, s_2(!v_2 \parallel s_3)$
  $\Longrightarrow \quad t = s_1 !v_1\, t'$ and $t' \in s_2(!v_2 \parallel s_3)$
  $\Longrightarrow \quad t = s_1 !v_1\, t'$ and $t' \in [\![f']\!][\![\Delta]\!]\rho$ by I
  $\Longrightarrow \quad s_1 !v_1\, t' \in [\![f]\!][\![\Delta]\!]\rho$ by Lemma 36
  $\Longrightarrow \quad t \in [\![f]\!][\![\Delta]\!]\rho$
  $\Longrightarrow \quad s_1 !v_1\, s_2(!v_2 \parallel s_3) \subseteq [\![f]\!][\![\Delta]\!]\rho$ □

Lemma 56. $\{\!|f|\!\}\{\!|\Delta|\!\}\rho = \{\!|f >\!x\!> let(x)|\!\}\{\!|\Delta|\!\}\rho$

Proof. It suffices to show that one side is a subset of the other.
($\Rightarrow$) easy
($\Leftarrow$) This is the interesting case of the Lemma.
If $t \in [\![f >\!x\!> let(x)]\!][\![\Delta]\!]\rho$ we must show that $t\backslash\tau \in \{\!|f|\!\}\{\!|\Delta|\!\}\rho$
We know that there exists $s \in [\![f]\!][\![\Delta]\!]\rho$ such that $t \in s \gg \lambda v.\{[v/x]\,!v\}_{\mathrm{p}}\backslash[v/x]$
therefore $t \in s \gg \lambda v.\{!v\}_{\mathrm{p}}$
We proceed by induction on the number of publications in s

- If s has no publications, then $t = s$, trivial
- $s = s_1\,!u\,s_2, \quad \bar{P}(s_1)$
  $\Longrightarrow \quad t \in s_1\,\tau((s_2 \gg \lambda v.\{!v\}_{\mathrm{p}}) \parallel \{!u\}_{\mathrm{p}})$
  Also, by Lemma 38, $\Delta, \Gamma \vdash f \overset{s_1!u}{\rightarrow}{}_{*} f'$ and $s_2 \in [\![f']\!][\![\Delta]\!]\rho$ $\quad (I)$
  We take two separate cases because of the prefix-closed set $\{!u\}_{\mathrm{p}}$
  - $t \in s_1\,\tau((s_2 \gg \lambda v.\{!v\}_{\mathrm{p}}) \parallel \varepsilon)$
    so there exists $t'' \in (s_2 \gg \lambda v.\{!v\}_{\mathrm{p}})$ such that $t \in s_1\,\tau\,t''$
    Then, it suffices to show that $(s_1\,t'')\backslash\tau \in \{\!|f|\!\}\{\!|\Delta|\!\}\rho$
    Since $s_2 \in [\![f']\!][\![\Delta]\!]\rho$ we know $t'' \in [\![f' >\!x\!> let(x)]\!][\![\Delta]\!]\rho$
    $\Longrightarrow \quad \exists\, t' \in [\![f']\!][\![\Delta]\!]\rho. \quad t''\backslash\tau = t'\backslash\tau$ by *IH* for s_2 $\quad (II)$
    $\Longrightarrow \quad s_1\,!u\,t' \in [\![f]\!][\![\Delta]\!]\rho$ by I and Lemma 36
    $\Longrightarrow \quad s_1(!u \parallel t') \subseteq [\![f]\!][\![\Delta]\!]\rho$ by Lemma 55
    $\Longrightarrow \quad s_1\,t'\,!u \in [\![f]\!][\![\Delta]\!]\rho$
    $\Longrightarrow \quad s_1\,t' \in [\![f]\!][\![\Delta]\!]\rho$ by Theorem 6
    $\Longrightarrow \quad (s_1\,t')\backslash\tau \in \{\!|f|\!\}\{\!|\Delta|\!\}\rho$
    $\Longrightarrow \quad (s_1\,t'')\backslash\tau \in \{\!|f|\!\}\{\!|\Delta|\!\}\rho$ by II
    which is what we needed to show
  - $t \in s_1\,\tau((s_2 \gg \lambda v.\{!v\}_{\mathrm{p}}) \parallel !u)$
    similar to the previous case $\square$