A Fu y Abstract May est n e ant cs or Concurrent b ects

A an Je rey
C I De au n versty
C ca o IL A
a e rey cs epau e u

Ju an at e
C G n versty o ussex
Br ton K
u anr co s susx ac u

ctober ••

Abstract

s paper prov es a u y abstract se ant cs or a var ant o t e concurrent ob ect ca cu us e e ne ay test n or concurrent ob ect co ponents an t en c arater se t us n a trace se ant cs nsp re by ML nteract on a ra s e a n resu t o t s paper s to s ow t at t e trace se ant cs s u y abstract or ay test n s s t e rst suc resu t or a concurrent ob ect an ua e

Introduction

Aba an Car e s ob ect ca cu us s a n a an ua e or nvest at n eatures o ob ect an ua es suc as encapsu ate state subtyp n an se var ab es Gor on an Han n a e concurrent eatures to t e ob ect ca cu us to pro uce t e concurrent ob ect ca cu us

r or wor on t e ob ect ca cu us as concentrate on t e operat ona be av our o ob ect sys te s an type syste s w c prov e type sa ety uarantees e c osest paper to ours s Gor on an ees s u y abstract se ant cs or t e utab e s n e t rea e ob ect ca cu us ere as been no wor on prov n u y abstract se ant cs or concurrent utab e ob ects

In t s paper we present t e rst u y abstract test n se ant cs or a var ant o Gor on an Han n s concurrent ob ect ca cu us w t out subtyp n e ac o subtyp n ere a or s a s p er presentat on o t e abe e trans t ons an traces but we ant c pate t at t e proo tec n ques use ere are robust enou to cater or subtyp n a so s se ant cs was nsp re by ML nteract on a ra s w c are a co on too or v sua s n nteract ons w t ob ect syste s

Interaction diagrams

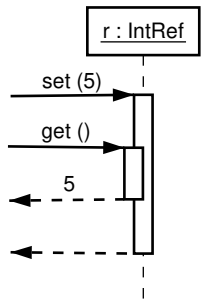
Interact on a ra s n part cu ar sequence a ra s were eve ope by Jacobson an are now part o t e n e Mo e n Lan ua e stan ar Interact on a ra s recor t e essa es sent between ob ects o a co ponent n an ob ect syste ese essa es nc u e et o ca s

esearc part a y supporte by t e u e Foun at on n versty o ussex tec n ca report ••

an returns nteract on a ra s nc u e ot er or s o essa e but we w not use t ese n t s paper

A s p e nteract on w t an nte er re erence ob ect r o type `IntRef`

equence a ra s can be use or u t t rea e app cat ons or exa p e



Here two t rea s n epen ent y ca et o s o t e ob ect r creat n a race con t on In our textua represent at on we ve t e t rea s na es an we ecorate eac essa e w t t e t rea respons b e or t e essa e

thread1 call r.set(?
thread2 call r.get

- Messa es are nco n or out o n essa e ca s or atc n out o n or nco n returns
- Messa es are ecorate w t t rea ent ers
- Messa es ay nc u e res na es

e ave on y use a very s a subset o sequence a ra s w c n turn s a very s a subset o ML but n t s paper we w s ow t at t s s a subset s very expressive an n part cu ar prov es a u y abstract se ant cs

The object calculus

e ob ect ca cu us s a n a an ua e or o e n ob ect base pro ra n Aba an Car e prov e a type syste an operat ona se ant cs or a var ety o ob ect ca cu an prove type sa ety or t e Gor on an Han n ave s nce exten e t s an ua e to nc u e concurrent eatures

In t s paper we s a nvest ate a var ant o Gor on an Han n s concurrent ob ect ca cu us w c nc u es

- A eap o na e ob ects an t rea s
- rea s can ca or up ate ob ect et o s can co pare ob ect or t rea na es or equa ty can create new ob ects an t rea s an can scover t e r own t rea na e
- An operat ona se ant cs base on t e π ca cu us an a s p e type syste
- A trace se ant cs as scusse n ect on

e are not cons er n any o t e ore a vance eatures o t e ob ect ca cu us or t e concurrent ob ect ca cu us suc as recurs ve types ob ect c on n an ob ect oc n s s ust or s p c ty an we o not see any tec n ca prob e s w t ncorporat n t ese eatures nto our an ua e

In anot er stran o researc D B as o an F s er a so es ne a ca cu us or o e n erat ve concurrent ob ect base syste s As w t Aba an Car e s ob ect ca cu us an ts var ous extens ons t e e p as s n D B as o an F s er s wor s a a n on type syste s an sa ety propert es or t e

Full abstraction

e prob e o u abstract on was rst ntro uce by M ner an nvest ate n ept by ot n Fu abstract on was rst propose or var ants o t e λ ca cu us but as s nce been nvest ate or process a ebras t e π ca cu us t e ν ca cu us Concurrent ML an t e utab e ob ect ca cu us

e can t en e ne t e *may testing preorder* as $C \sqsubseteq_{\text{ay}} C$ w enever

for any appropr ate y type C
$C \ \ C$ s success u t en $C \ \ C$ s success u

n ortunate y a t ou t s very s p e to e ne an s qu te ntu ve ay test n s o ten very cu t to reason about rect y because o t e quant cat on over any appropr ate y type C In pract ce we requ re a proo tec n que w c we can use to s ow resu ts about ay test n

ne approac s to use a *trace* se ant cs ven by e n n poss b e execut ons o co ponents $C \xrightarrow{s} C'$ w ere s s a sequence o essa es e t en wr te $\text{Traces}(C$ or t e set o a traces o C e say t at

- races are *sound* or ay test n w en $\text{Traces}(C \ \ \text{Traces}(C \ \ p es\ C \sqsubseteq_{\text{ay}} C$
- races are *complete* or ay test n w en $C \sqsubseteq_{\text{ay}} C \ \ p es\ \text{Traces}(C \ \ \text{Traces}(C$
- races are *fully absdwhereTj11.993tTJ/R1.991Tf1.19Tdfor-2may-22.9test-21.9ing-2.whenTJsi.9911.so/R11.9*

Components	C	$= \ldots \mid C \mid C \mid \nu(n\ T).C \mid n\ O \mid n(t)$
objects	O	$= l = M, \ldots, l = M$
Methods	M	$= \varsigma(n\ T).\lambda(x\ T, \ldots, x\ T).(t)$
areas	t	$= v \mid \text{stop} \mid \text{let } x\ T = e \text{ in } t$

e n.t.on o. e s *f* as zero ar u ent et o s

- A e ec arat.on $f = v$ .n an ob ect .s syntax su ar. or a et o ec arat.on $f = \varsigma(n$ T .λ(.□v□
- A e type *f* T .n an ob ect type .s syntax su ar. or a et o type *f* (→ T
- A e access express.on $v.f$ .s syntax su ar. or a et o ca $v.f($
- A e up ate express.on *n.f* $= v$□ .s syntax su ar. or a et o up ate $n.f$ □ (ς(*p* T .λ(. □*v*□

In a .t.on we ave restr.cte any subexpress.ons o. an express.on to be va ues rat er t an .u express.ons .or exa p e .n a et o ca $v.l(\vec{v}$ we requ.re t e ob ect an t e ar u ents to be va ues rat er t an express.ons $e.l(\vec{e}$ .s a es t e operat.ona se ant.cs uc eas.er to e ne an oes not restr.ct t e express.v.ty o. t e an ua e .or exa p e we can e ne ($e.l(\vec{e}$ ≡ (let $x =$ e in let $\vec{x} = \vec{e}$ in $x.l(\vec{x}$ .. ar y t e .st.nct.on between t rea s an express.ons a es t e

A t rea t cons sts o a stac o et express ons ter nate e t er by a return va ue

$$\text{let } x = T = e \text{ in } \cdots \text{let } x_n = T_n = e_n \text{ in } v$$

or by a ea oc e stop t rea

$$\text{let } x = T = e \text{ in } \cdots \text{let } x_n = T_n = e_n \text{ in stop}$$

Eac express on s e t er tse a t rea or

- an express on if $v = v$ then e else e
- a et o ca $v.l(\vec{v}$
- a et o up ate $n.l \Leftarrow M$ on a na e ob ect
- a new ob ect new O
- a new t rea new t or
- t e current t rea na e currentthread

Eac va ue s s p y a na e or a var ab e an we e er t e scuss on o types unt ect on

. Static semantics

e stat c se ant cs or our concurrent ob ect ca cu us s ven n F ures Most o t e ru es are stra t orwar a aptat ons o t ose ven by Aba an Car e e a n u e ent s $\Delta \vdash C : \Theta$ w c s rea as t e co ponent C uses na es Δ an e nes na es Θ For exa p e we e ne C (v C an IntRef as

$$\begin{array}{l} C \equiv \\ \quad \text{let } x = p.\text{get}(\text{ in } p.\text{set}(x \quad , \text{stop} \\ \\ \text{IntRef} \equiv \\ \quad \text{contents} : \text{Int} \end{array}$$

$$\frac{}{\Delta \vdash \top <: (} \qquad \frac{, \Delta, n\ T \vdash O\ T}{\Delta \vdash n O\ (n\ T} \qquad \frac{, \Delta, n\ \text{thread} \vdash t\ \text{none}}{\Delta \vdash n \vdash t\ (n\ \text{thread}}$$

$$\frac{\Delta, \Theta\ \vdash C\ \Theta \quad \Delta, \Theta\ \vdash C\ \Theta}{\Delta \vdash (C\ \vdash C\ (\Theta\ , \Theta} \qquad \frac{\Delta \vdash C\ \Theta, n\ T}{\Delta \vdash \forall (n\ T\ . C\ \Theta}$$

F ure u es or u e ent $\Delta \vdash C\ \Theta$

$$\frac{\Gamma, \Delta \vdash M\ T.l \quad \cdots \quad \Gamma, \Delta \vdash M_k\ T.l_k}{\Gamma, \Delta \vdash\ l\ = M\ , \ldots, l_k = M_k\ T}$$

F ure u e or u e ent $\Gamma, \Delta \vdash O\ T$ w en $T = l\ L\ , \ldots, l_k\ L_k$

$$\frac{\Gamma, x\ T\ , \ldots, x_k\ T_k, \Delta, n\ T \vdash t\ U}{\Gamma, \Delta \vdash \varsigma (n\ T\ . \lambda (x\ T\ , \ldots, x_k\ T_k\ . t\ T.l}$$

F ure u e or u e ent $\Gamma, \Delta \vdash M\ T.l$ w en $T = \ldots, l\ (T\ , \ldots, T_k \to U, \ldots$ an $T.l$ s t e recor l se ecte ro T

$$\frac{\begin{array}{c}\Gamma, \Delta \vdash v\ T \quad \Gamma, \Delta \vdash v\ T \\ \Gamma, \Delta \vdash e\ T \quad \Gamma, \Delta \vdash e\ T\end{array}}{\Gamma, \Delta \vdash \text{if } v\ = v\ \text{ then } e\ \text{ else } e\ T}$$

$$\frac{\begin{array}{c}\Gamma, \Delta \vdash v\ \ldots, l\ (T\ , \ldots, T_k \to T, \ldots \\ \Gamma, \Delta \vdash v\ T \quad \cdots \quad \Gamma, \Delta \vdash v_k\ T_k\end{array}}{\Gamma, \Delta \vdash v.l(v\ , \ldots, v_k\ T} \qquad \frac{\Gamma, \Delta \vdash n\ T \quad \Gamma, \Delta \vdash M\ T.l}{\Gamma, \Delta \vdash n.l \vdash M\ T}$$

$$\frac{\Gamma, \Delta \vdash O\ T}{\Gamma, \Delta \vdash \text{new } O\ T} \qquad \frac{\Gamma, \Delta \vdash t\ T}{\Gamma, \Delta \vdash \text{new} \vdash t \vdash\ \text{thread}} \qquad \frac{}{\Gamma, \Delta \vdash \text{currentthread}\ \text{thread}}$$

$$\frac{\Gamma, \Delta \vdash e\ T \quad \Gamma, x\ T\ , \Delta \vdash t\ T}{\Gamma, \Delta \vdash \text{let } x\ T\ = e \text{ in } t\ T} \qquad \frac{}{\Gamma, \Delta \vdash \text{stop}\ T} \qquad \frac{}{\Gamma, x\ T, \Gamma', \Delta \vdash x\ T} \qquad \frac{}{\Gamma, \Delta, n\ T, \Delta' \vdash n\ T}$$

F ure u es or u e ent $\Gamma, \Delta \vdash e\ T$

ar ab e contexts $\Gamma = x\ T, \ldots, x\ T$ a e contexts $\Delta, \Theta, \Sigma, \Phi = n\ T, \ldots, n\ T$

In var ab e contexts var ab es ust be un que an are v ewe up to reor er n
In na e contexts na es ust be un que types ust not be none an are v ewe up to reor er n

F ure yntax o na e an var ab e contexts

enever $\Delta \vdash C \Theta$ conta ns a subexpress on o n appears n Θ

s s nten e to capture t e co on so tware en n s ou ot export utab e e s nstea t ey s ou export su e t e on urat ons C an C above are wr te c ose s nce t ow n o ponent w c wr tes rect y to p.contents s not w

$$C' \equiv n[\text{let } x = p.\text{contents in } p.\text{co}$$

or t e re an er o t e paper we w requ re co po e op n a u y abstract se ant cs uc s p er s nc rect y

Dynamic semantics

e yna c se ant cs or our concurrent ob ect ca cu n n e e ne t ree re at ons between co ponents

- $\equiv$ structura con ruence represents t e east con n co ax o s n F ure

t

$C \equiv C \quad (C \quad C \quad C \equiv C \quad (C \quad C \quad C \quad C \equiv C \quad C$

$C \quad \nu(n \quad T \quad .C \equiv \nu(n \quad T \quad .(C \quad C \quad \nu(n \quad T \quad .\nu(n \quad T \quad .C \equiv \nu(n \quad T \quad .\nu(n \quad T \quad .C$

Figure Axioms for structural congruence where n is not free in C

$$n(\text{let } x \quad T = v \text{ in } t) \xrightarrow{\beta} n(t\ v/x)$$

$$n(\text{let } x \quad T = (\text{let } x \quad T = e \quad \text{in } e \quad \text{in } t) \xrightarrow{\beta} n(\text{let } x \quad T = e \quad \text{in } (\text{let } x \quad T = e \quad \text{in } t)$$

$$n(\text{let } x \quad T = (\text{if } v \quad = v \text{ then } e \text{ else } e \quad \text{in } t) \xrightarrow{\beta} n(\text{let } x \quad T = e \quad \text{in } t)$$

$$n(\text{let } x \quad T = (\text{if } v \quad = v \quad \text{then } e \quad \text{else } e \quad \text{in } t) \xrightarrow{\beta} n(\text{let } x \quad T = e \quad \text{in } t) \qquad v \quad v$$

$$n(\text{let } x \quad T = \text{new } O \text{ in } t) \xrightarrow{\beta} \nu(p \quad T \quad .(p\ O \quad n(\text{let } x \quad T = p \text{ in } t) \qquad \text{else}$$

$$(\text{let } x \quad T =$$

. Testing preorder

e w now e ne t e test n se ant cs or our concurrent ob ect ca cu us e w o t s by e n n a not on o *barb*

(Δ, *n* thread □ *C*

then where C (v is defined in Section ...) we have

$$(\,C\,(\;\Theta$$

$$\xrightarrow{\nu(n\ \mathsf{thread}\ .n\ \mathsf{call}\ p.\mathsf{get}(\,)?)}$$

$$(\,(C\,(\;\;n\ \mathrm{let}\ x = p.\mathrm{get}(\,)\ \mathrm{in\ return}\ x\;\;\Theta'$$

$$(\,(C\,(\;\;n\ \mathrm{return}\;\;\Theta'$$

$$\xrightarrow{n\ \mathsf{return}}$$

$$(\,(C\,(\;\;n\ \mathrm{block}\;\;\Theta'$$

$$\xrightarrow{n\ \mathsf{call}\ p.\mathsf{set}(\,)?}$$

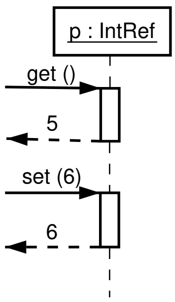
$$(\,(C\,(\;\;n\ \mathrm{let}\ x = p.\mathrm{set}(\,)\ \mathrm{in\ return}\ x\;\;\Theta'$$

$$(\,(C\,(\;\;n\ \mathrm{return}\;\;\Theta'$$

$$\xrightarrow{n\ \mathsf{return}}$$

$$(\,(C\,(\;\;n\ \mathrm{block}\;\;\Theta'$$

which corresponds to the interaction diagram



For any component $(\Delta\ C\ \Theta)$ we define its traces to be

$$\mathrm{Traces}(\Delta\ C\ \Theta) = \{s \mid (\Delta\ C\ \Theta) \stackrel{s}{\Longrightarrow} (\Delta'\ C' \ldots$$

t e base an ua e w c wou ave been reac e a t e co ponent an test actua y nteracte
s operat on o er n s e ne be ow

. The merge operator

De ne t e part a *merge* operator $C \between C$ on co ponents as t e sy etr c operator e ne up to $\equiv$ w ere

$$
\begin{aligned}
\mathbf{0} \between C &= C \\
(\nu(p\ T)\,.C') \between C &= \nu(p\ T)\,.(C' \between C) \\
(p\ O\ \Box)C \between C &= p\ O\ \Box(C \between C) \\
(p\Box t\Box\Box)C \between C &= p\Box t\Box\Box(C \between C) \\
(n\Box t\ \Box\Box)C \between (n\Box t\ \Box\Box)C &= n\Box t\ \between t\ \Box\Box(C \between C)
\end{aligned}
$$

w en $n \Box \mathrm{dom}(C, C)$ an $p \Box \mathrm{fn}(C)$

e over oa notat on an e ne t e part a er e operator $t \between t$ on t rea s as t e sy etr c operator w ere

$$
\begin{aligned}
(\mathrm{let}\ x\ T = \mathrm{block}\ \mathrm{in}\ t) \between \mathrm{stop} &= \mathrm{stop} \\
(\mathrm{let}\ x\ T = \mathrm{block}\ \mathrm{in}\ t) \between (\mathrm{let}\ y\ U = \mathrm{return}\ v\ T\ \mathrm{in}\ t) &= (\mathrm{let}\ y\ U = \mathrm{block}\ \mathrm{in}\ t \between (t\ v/x) \\
(\mathrm{let}\ x\ T = \mathrm{block}\ \mathrm{in}\ t) \between (\mathrm{let}\ y\ U = e\ \mathrm{in}\ t) &= \mathrm{let}\ y\ U = e\ \mathrm{in}\ ((\mathrm{let}\ x\ T = \mathrm{block}\ \mathrm{in}\ t) \between t)
\end{aligned}
$$

w en e s b oc return ree an $y \Box \mathrm{fv}(t)$

Lemma . *If* $\Delta \Box (C\ \Box C)\ \Theta$ *then* $(C \between C \equiv (C\ \Box C)$.

Proof An n uct on on t e e n t on o $C \between C$ □

Lemma . *If* $C \between C \equiv C$ *and* $C \downarrow_b$ *then* $C\downarrow_b$.

Proof An n uct on on t e e n t on o $C \between C$ □

. Trace composition and decomposition

G ven a trace s we wr te s or t e co p e entary trace

$\varepsilon = \varepsilon\ C$ An vBI IMtrue H B C ID EIe been

Proof Given in Appendix A. □

Corollary . *For any components* $(\Delta, \Phi \vdash C \multimap \Theta, \Sigma$ *and* $(\Theta, \Phi \vdash C \multimap \Delta, \Sigma$ *such that* $C \between C \equiv C$ *and* $C\downarrow_b$ *then there exists some trace* s *such that* $(\Delta, \Phi \vdash C \multimap \Theta, \Sigma \stackrel{s}{\Longrightarrow} (\Delta', \Phi \vdash C' \multimap \Theta', \Sigma'$ *and* $(\Theta, \Phi \vdash C \multimap \Delta, \Sigma \stackrel{s}{\Longrightarrow} (\Theta', \Phi \vdash C' \multimap \Delta', \Sigma'$ *where either* $C'\downarrow_b$ *or* $C'\downarrow_b$.

Proof We now that $C\downarrow_b$ we use to us that $C \equiv C''$ for some C'' such that $C''\downarrow_b$ We use Proposition to obtain a trace s such that

$$(\Delta, \Phi \vdash C \multimap \Theta, \Sigma \stackrel{s}{\Longrightarrow} (\Delta'', \Phi \vdash C'' \multimap \Theta'', \Sigma''$$
$$(\Theta, \Phi \vdash C \multimap \Delta, \Sigma \stackrel{s}{\Longrightarrow} (\Theta'', \Phi \vdash C'' \multimap \Delta'', \Sigma''$$

where $\nu(\Delta'', \Theta'', \Sigma'') \setminus \Delta, \Theta, \Sigma$. $(C'' \between C'' \equiv C''$. Given that $C''\downarrow_b$ we now that $(C'' \between C'')\downarrow_b$ also By the definition of $\between$ we see that one of the following or their symmetric counterparts must hold

- $C''\downarrow_b$ and we are done or
- $C'' \equiv \nu(\Delta$. $(n[t \mid\mid C'''$ and $C'' \equiv \nu(\Delta$. $(n[t \mid\mid C'''$ where $n[t \between t\downarrow_b$ We now proceed by induction on the definition of $t \between t$ to show that for all such C'' and C'' we can find s where
$$(\Delta'', \Phi \vdash C'' \multimap \Theta'', \Sigma'' \stackrel{s}{\Longrightarrow} (\Delta', \Phi \vdash C' \multimap \Theta', \Sigma'$$
$$(\Theta'', \Phi \vdash C'' \multimap \Delta'', \Sigma'' \stackrel{s}{\Longrightarrow} (\Theta', \Phi \vdash C' \multimap \Delta', \Sigma'$$
and either $C'\downarrow_b$ or $C'\downarrow_b$ There are two cases up to symmetry of $\between$
  - If $t = \text{let } x = T = \text{block in } t''$ and $t = \text{let } y = U = b.\text{succ}(\ )$ in t'' then $C''\downarrow_b$
  - If $t = \text{let } x = T = \text{block in } t''$ and $t = \text{let } y = U = \text{return}(v = T$ in t'' then we have
$$(\Delta'', \Phi \vdash C'' \multimap \Theta'', \Sigma'' \xrightarrow{\nu(\Delta' . n[\text{return } v]?} (\Delta'', \Delta', \Phi \vdash \nu(\Delta . (n[t'' \mid v/x \mid\mid C''' \multimap \Theta', \Sigma'$$
$$(\Theta'', \Phi \vdash C'' \multimap \Delta'', \Sigma'' \xrightarrow{\nu(\Delta' . n[\text{return } v]} (\Theta'', \Phi \vdash \nu(\Delta'' . (n[\text{let } y = U = \text{block in } t'' \mid\mid C''' \multimap \Delta'', \Delta', \Sigma''$$
where $\Delta = (\Delta', \Delta''$ and moreover
$$n[t \between t \mid \equiv n[(\text{let } y = U = \text{block in } t'' \between t \mid v/x \mid\downarrow_b$$
so by inductive hypothesis
$$(\Delta'', \Phi \vdash C'' \multimap \Theta'', \Sigma'' \xrightarrow{\nu(\Delta' . n[\text{return } v]?} \stackrel{s}{\Longrightarrow} (\Delta', \Phi \vdash C' \multimap \Theta', \Sigma'$$
$$(\Theta'', \Phi \vdash C'' \multimap \Delta'', \Sigma'' \xrightarrow{\nu(\Delta' . n[\text{return } v]} \stackrel{s}{\Longrightarrow} (\Theta', \Phi \vdash C' \multimap \Delta', \Sigma'$$
and either $C'\downarrow_b$ or $C'\downarrow_b$ as required □

. Proof of soundness

Theorem . **(Soundness of traces for may testing)** *If* Traces(Δ C Θ Traces(Δ C Θ *then* Δ ⊨ *C* ⊑*may* *C* Θ

Proof uppose t at Traces(Δ C Θ Traces(Δ C Θ an t at we ave (Θ,*b* barb C Δ suc t at (*C* *C* *b*, we ust s ow t at (*C* *C* *b* a so
ow s nce (*C* *C* *b* we can use Coro ary to et

$$(\Delta, b \text{ barb } \square C \ \Theta \xrightarrow{s} \square\ (\Delta', b \text{ barb } \square C' \ \Theta', \Sigma'$$

$$(\Theta, b \text{ barb } \square C \ \Delta \xrightarrow{s} \square\ (\Theta', b \text{ barb } \square C' \ \Delta', \Sigma'$$

$$\overline{\Delta \square \varepsilon \quad \text{trace } \Theta}$$

n

- If $n \in$ threads $(s$ then n.balance $=$ in s
- If n.balance $=$ in s and s then n.balance $=$ in s
- If n.balance $=$ in s then n.balance $=$ in $\nu(\Delta$. n!call $p.l(\vec{n}) ? s \nu(\Theta$. n?return $v)$
- If n.balance $=$ in s then n.balance $=$ in $\nu(\Theta$. n!call $p.l(\vec{n}) \; s \nu(\Delta$. n?return $v)$?

Define pop$n(s$—as

- If n.balance $=$ in s then pop$n(s$ $=$
- If n.balance $=$ in s and $a =$

Proof Easy ..n uct..on on *s*' □

Lemma .ſ

1. *If C is block/return free and* $(\Delta \vdash C\ \Theta \overset{s}{=\!=} \square \xrightarrow{\nu(\Theta'.n\square\text{return } v\square)} \square$ *then* $s = s\ \nu(\Delta'.n\square\text{call } p.l(\vec{v}\ \square?\ s$ *where n is balanced in s .*

2. *If C is block/return free and* $(\Delta \vdash C\ \Theta \overset{s}{=\!=} \square \xrightarrow{\nu(\Delta'.n\square\text{return } v\square?} \square$ *then* $s = s\ \nu(\Theta'.n\square\text{call } p.l(\vec{v}\ \square\ s$ *where n is balanced in s .*

Proof e prove t ese propert..es s.. u taneous y by an ..n uct..on on t e en t o.. *s* e on y s ow t e ar u ent.. or art as art can be s own ..n a s.. ..ar anner By ana ys..s o.. t e ru es o.. t e ts we ave

$$(\Delta \vdash C\ \Theta \overset{s}{=\!=} \square (\Delta'' \vdash C''\ n\square\text{let } x\ T = \text{return } (v\ U\ \text{in } t\square\ \Theta''\ \xrightarrow{\nu(\Theta'.n\square\text{return } v\square)}\rightarrow$$

ow part..t..on *s* ..nto *s s* p..c ..n *s*

Case $s = s' \nu(\Delta'').n!\mathrm{call}\, p.l(\vec{v}).?$ We now that

$$(\Delta \vdash C) \Theta \xrightarrow{s'} (\Delta, \Delta(s') \vdash C') \Theta, \Theta(s') \xrightarrow{\nu(\Delta'').n!\mathrm{call}\, p.l(\vec{v}).?}$$

so we have that either

$$C' \equiv \nu(\Delta'').\nu(\Delta''').n!\mathrm{let}\, x = T = \mathrm{block\ in}\ t \mid C''$$

or $n \in \Delta, \Delta(s')$ and n is a first thread to s'. We can apply the induction hypothesis to see that $\Delta \vdash s'$ trace Θ and we consider $\mathrm{pop}\, n(s')$. If $n \in \Delta, \Delta(s')$ and n is first thread to s' then $\mathrm{pop}\, n(s')$ is necessarily ε. Otherwise we know that $C' \equiv \nu(\Delta'').\nu(\Delta''').n!\mathrm{let}\, x = T = \mathrm{block\ in}\ t \mid C''$ and therefore the last action which could have occurred at n must have been an output that is $\mathrm{pop}\, n(s') = \gamma$. In both cases we see that

$$n \text{ is input enabled in } \Delta \vdash s' \text{ trace } \Theta$$

We know that $(\Delta, \Delta(s') \vdash C') \Theta, \Theta(s') \xrightarrow{\nu(\Delta'').n!\mathrm{call}\, p.l(\vec{v}).?}$ and we know that the session conditions on the transition rule for $\nu(\Delta'').\gamma.?$ actions guarantees that

$$\mathrm{dom}(\Delta'') \cap \mathrm{fn}(\vec{v})$$

We also know that the session conditions on rule for call inputs actions guarantees that

$$, \Delta, \Delta(s') \vdash , \Theta, \Theta(s') \vdash , \Delta' \vdash p.l(\vec{v}) : T \text{ and } p \in \Theta, \Theta(s')$$

We use this to see that

$$, \Theta, \Theta(s') \vdash , \Delta' \vdash p : \&\{l : (\vec{T}) \to T\}$$

and

$$, \Delta, \Delta(s') \vdash , \Theta, \Theta(s') \vdash , \Delta' \vdash \vec{v} : \vec{T}$$

Lastly it is easy to see that

$$, \Delta, \Delta(s') \vdash , \Theta, \Theta(s') \vdash , \Delta' \vdash n : \mathrm{thread}$$

We collect the statements together to see that they form the hypotheses of the type rule which allows us to conclude

$$\Delta \vdash s' \nu(\Delta'').n!\mathrm{call}\, p.l(\vec{v}).? \text{ trace } \Theta$$

as required.

Case $s = s' \nu(\Theta'').n!\mathrm{call}\, p.l(\vec{v})$. Similar to previous case.

Case $s = s' \nu(\Theta'').n!\mathrm{return}\, v$. We now that

$$(\Delta \vdash C$$

$\Delta \vdash s'$ trace Θ

an we not ce t at because C s b oc return ree we can app y Le a to et

$$s' = s\ \nu(\Delta'\ .\ n!\text{call}\ p.l(\vec{v}\ ?\ s$$

w ere n s ba ance n s G ven t s we see t at

$$\text{pop}\, n(s\ \nu(\Delta'\ .\ n!\text{call}\ p.l(\vec{v}\ ?\ s\ = \nu(\Delta'\ .\ n!\text{call}\ p.l(\vec{v}\ ?$$

ence

$$\text{pop}\, n(s' = \nu(\Delta'\ .\ n!\text{call}\ p.l(\vec{v}\ ?$$

A a n t e s e con t ons on t e trans t on ru e or $\nu(\Theta'\ .\ \gamma$ uarantee t at

$$\text{dom}(\Theta' \subseteq \text{fn}(v$$

e a so now by an t e act t at pre xes o we type traces are a so we type t at

$$\Delta \vdash s\ \nu(\Delta'\ .\ n!\text{call}\ p.l(\vec{v}\ ?\ \text{trace}\ \Theta$$

an we see t at t s ust ave been n erre us n a ypot es s

$$,\Theta,\Theta(s\ \vdash p\ \ldots l\ (\vec{U}\ \to U'\ldots$$

w c by wea en n ves us

$$,\Theta,\Theta(s'\ \vdash p\ \ldots l\ (\vec{U}\ \to U'\ldots$$

Last y because

$$(\Delta,\Delta(s'\ \vdash C'\ \Theta,\Theta(s'$$

an

$$C' \equiv C'\ n!\text{let}\ x\ T = \text{return}\ (v\ U\ \text{in}\ t!$$

we see t at

$$,\Delta,\Delta(s'\ ,\Theta,\Theta(s'\ ,\Theta'\ \vdash v\ U$$

o by Le a to et er w t t e typ n s e con t ons or ca nput trans t ons we ave t at $U = U'$ an so

$$,\Delta,\Delta(s\ ,\Theta,\Theta(s\ ,\Theta'\ \vdash v\ U$$

e co ect t e state ents to et er to see t at t ey or t e ypot eses o t e type ru e w c a ows us to conc u e

$$\Delta \vdash s'\ \nu(\Theta'\ .\ n!\text{return}\ v!\ \text{trace}\ \Theta$$

as requ re

Case $s = s'\ \nu(\Delta'\ .\ n!\text{return}\ v!?$ ar to prev ous case □

. Information order on traces

We define an information preorder on traces $\Delta \vdash r \sqsubseteq s$ trace Θ as the least relation generated by axioms such that each case we require both sides of the inequation to be well-typed traces:

$$\Delta \vdash s \sqsubseteq sr \text{ trace } \Theta$$
$$\Delta \vdash s\gamma? \sqsubseteq s \text{ trace } \Theta$$
$$\Delta \vdash s\gamma\ ?\gamma \quad r \vdash s\gamma \quad \gamma \sqsubseteq ?r \text{ trace } \Theta$$
$$\Delta \vdash s\nu(\Delta\ .\gamma\ ?\gamma\ ?r \vdash s\nu(\Delta\ .\gamma\ ?\gamma \sqsubseteq ?r \text{ trace } \Theta$$
$$\Delta \vdash s\nu(\Theta\ .\gamma\ \gamma\ \ r \vdash s\nu(\Theta\ .\gamma\ \ \gamma \sqsubseteq r \text{ trace } \Theta$$

Lemma . (Information Order Duality) *If* $\Delta \vdash r\gamma\ \ \sqsubseteq s\gamma$ trace Θ *and* fn $(\gamma\ \cap \Theta(r\ = \emptyset$ *and* $\gamma\ \ \sharp s, r$ *then* $\Theta \vdash s \sqsubseteq r$ trace Δ.

Proof We write $\Delta \vdash r \sqsubseteq^{n} s$ trace Θ for $\Delta \vdash r \sqsubseteq s$ trace Θ

Proposition . (Information Order Closure) *If* $(\Delta \sqsubseteq C\ \Theta \stackrel{s}{=\!=} \sqsubseteq$ *and* $\Delta \sqsubseteq r \sqsubseteq s$ trace Θ *then* $(\Delta \sqsubseteq C\ \Theta \stackrel{r}{=\!=} \sqsubseteq$.

Proof ow t at t e o ow n a ra s can be co p ete w en thread $(\gamma$ ⊟ thread $(\gamma$

.

```
Comp (Δ□ s  trace Θ  = ν(Θ(s ,ref  Ref, state_ε  State  .(
   ref val = state_ε  □
   state_ε State(Δ □ ε ≤ s  trace Θ  □
   ∏{p l_i = ref.val.inCall_{p.l_i L_i} | i =  ...n ┼ p— l_i  L_i | i =  ...n  □ Θ, Θ(s }□
   ∏{n□ref.val.out_none( □┼ n  thread □ Θ, Θ(s }

Ref = val  State
State =  out_T  (  → T, inRetu rn_T  (T  → T, inCall_{p.l L}  L
State(Δ □ r ≤ s  trace Θ  = (
   out_T = Out_T(Δ □ r ≤ s  trace Θ ,
   inReturn_T = InReturn_T(Δ □ r ≤ s  trace Θ ,
   inCall_{p.l L} = InCall_{p.l L}(Δ □ r ≤ s  trace Θ

Out_T(Δ □ r ≤ s  trace Θ  = λ(  .(
   w en r a ≤ s an  a = ν(Θ□  . n□call p.l(v⃗ □  an  ,Δ, Θ, Δ(r  , Θ(r  , Θ□ □ p.l(v⃗ —U
      if currentthread = n then
         ref.val  = new State(Δ □ r a ≤ s  trace Θ  ,
         ref.val.inReturn_U(p.l(v⃗  ,
         ref.val.out_T(
   w en r a ≤ s an  a = ν(Θ□  . n□return v□  an  ,Δ, Θ, Δ(r  , Θ(r  , Θ□ □ v— T
      if currentthread = n then
         ref.val  = new State(Δ □ r a ≤ s  trace Θ  ,
         v
   ot erw se
      stop

InReturn_T(Δ □ r ≤ s  trace Θ  = λ(x  T  .(
   w en r a ≤ s an  a = ν(Δ□  . n□return v□? an  ,Δ, Θ, Δ(r  , Θ(r  , Δ□ □ v— T
      if Δ, Θ, Δ(r  , Θ(r  □ (currentthread, x  = ν(Δ□  .(n, v  then
         ref.val  = new State(Δ □ r a ≤ s  trace Θ  ,
         v
   ot erw se
      stop

InCall_{p.l (T⃗ → T}(Δ □ r ≤ s  trace Θ  = λ(x⃗  T⃗  .(
   w en r a ≤ s an  a = ν(Δ□  . n□call p.l(v⃗ □? an  ,Δ, Θ, Δ(r  , Θ(r  , Δ□ □ v⃗— T⃗
      if Δ, Θ, Δ(r  , Θ(r  □ (currentthread, x⃗  = ν(Δ□  .(n, v⃗  then
         ref.val  = new State(Δ □ r a ≤ s  trace Θ  ,
         ref.val.out_T(
   ot erw se
      stop

F ure—  De n t on o  Comp (Δ□ s  trace Θ
```

$$\text{if } \Delta \square (\quad = \nu(\quad .(\quad \text{then } t \quad = \quad t$$

$$\text{if } \Delta \square (v, \vec{v} \quad = \nu(p \quad U, \vec{n} \quad \vec{T} \quad .(p, \vec{p} \quad \text{then } t \quad = \quad \text{if } v \square \Delta^{-} \ (U \quad \text{then}$$

$$(\text{if } \Delta, p \quad U \square (\vec{v} \quad = \nu(\vec{n} \quad \vec{T} \quad .(\vec{p} \quad \text{then } t \quad v/p \quad \text{else stop}$$

$$\text{if } \Delta \square (v, \vec{\ }$$

. Proof of completeness

Theorem . . (Completeness of traces for may testing) *If* $\Delta \models C \sqsubseteq_{may} \Theta$ *then* Traces($\Delta \| C$) $\subseteq$ Traces($\Delta \| C$).

Proof Choose any trace $s \in$ Traces

1. *If* $C \equiv C \equiv D \mid E$ *then there exist components such that* $C \equiv D \mid E$ *and* $C \equiv D \mid E$ *with* $D \equiv D \equiv D$ *and* $E \equiv E \equiv E$.

2. *If* $C \equiv C \equiv \nu(\vec{n})\ \vec{T}\ .C$ *then there exist components such that* $C \equiv \nu(\vec{n})\ \vec{T}\ .C'$ *and* $C \equiv \nu(\vec{n})\ \vec{T}\ .C'$ *with* $(\vec{n})\ \vec{T} = (\vec{n})\ \vec{T}, \vec{n})\ \vec{T}$ *and* $C' \equiv C' \equiv C'$.

Proof rove by n uct on on t e er vat on o $C \equiv C$ □

Lemma A. *If* $C \equiv C \equiv C$ *and* $C \xrightarrow{\beta} C'$b

- **Case** ($\gamma = \nu(\vec{n}\ \vec{T})\ .\ n\triangleleft \mathrm{call}\ p.l(\vec{v}\triangleright$ an $n \in \Sigma$:

  ar to t e prev ous case

- **Case** ($\gamma = \nu(\vec{n}\ \vec{T})\ .\ n\triangleleft \mathrm{return}\ v\triangleright$:

  nce $(\Delta, \Phi \vdash C \Theta, \Sigma \xrightarrow{\gamma'} (\Delta', \Phi \vdash C' \Theta', \Sigma'$ we ust ave t at

  $$C \equiv \nu(\vec{p}\ \vec{U})\ .(C'' \mid n\triangleright \mathrm{let}\ x\ T = \mathrm{block}\ \mathrm{in}\ t \triangleleft$$
  $$C' \equiv \nu(\vec{p}\ \vec{U})\ .(C'' \mid n\triangleright t\ \ v/x \triangleleft$$
  $$\Delta' = \Delta, \vec{n}\ \vec{T}$$
  $$\Theta' = \Theta$$
  $$\Sigma' = \Sigma$$

  nce $(\Theta, \Phi \vdash C \Delta, \Sigma \xrightarrow{\gamma} (\Theta', \Phi \vdash C' \Delta', \Sigma'$ we ust ave t at

  $$C \equiv \nu(\vec{n}\ \vec{T})\ .\nu(\vec{p}\ \vec{U})\ .(C'' \mid n\triangleright \mathrm{let}\ y\ U = \mathrm{return}\ (v\ T)\ \mathrm{in}\ t \triangleleft$$
  $$C' \equiv \nu(\vec{p}\ \vec{U})\ .(C'' \mid n\triangleright \mathrm{let}\ y\ U = \mathrm{block}\ \mathrm{in}\ t \triangleleft$$

  e t en s ow t at

  $$C \between C \equiv \nu(\vec{n}\ \vec{T})\ .\nu(\vec{p}\ \vec{U})\ .\nu(\vec{p}\ \vec{U})\ .((C'' \between C'' \triangleright n\triangleright(\mathrm{let}\ y\ U = \mathrm{block}\ \mathrm{in}\ t \between (t\ \ v/x \triangleleft$$

  an t at

  $$C' \between C' \equiv \nu(\vec{p}\ \vec{U})\ .\nu(\vec{p}\ \vec{U})\ .((C'' \between C'' \triangleright n\triangleright(\mathrm{let}\ y\ U = \mathrm{block}\ \mathrm{in}\ t \between (t\ \ v/x \triangleleft$$

  an so

  $$\nu(\Delta', \Theta', \Sigma' \setminus \Delta, \Theta, \Sigma\ .(C' \between C' \equiv C$$

  as requ re : □

Co pos t on o ows by n uct on on t e er vat on o $(\Delta, \Phi \vdash C \Theta, \Sigma \xrightarrow{s} (\Delta', \Phi \vdash C' \Theta', \Sigma'$ an $(\Theta, \Phi \vdash C \Delta, \Sigma \xrightarrow{s} (\Theta', \Phi \vdash C' \Delta', \Sigma'$ a n use o Le as A A an A :

A. Decomposition

e s ow t ree e as ro w c Deco pos t on o ows:

Lemma A. *For any* $\Delta, \Phi \vdash C \Theta, \Sigma$ *and* $\Theta, \Phi \vdash C \Delta, \Sigma$ *if* $(C \between C \equiv \nu(\vec{n}\ \vec{T})\ .(C \triangleright n\triangleright \mathrm{let}\ x\ T = e\ \mathrm{in}\ t \triangleleft$ *then either we have:*

$$(\Delta, \Phi \vdash C \Theta, \Sigma \xrightarrow{s} (\Delta', \Phi \vdash \nu(\vec{n}\ \vec{T})\ .(C' \triangleright n\triangleright \mathrm{let}\ x\ T = e\ \mathrm{in}\ t \triangleleft \Theta', \Sigma'$$
$$(\Theta, \Phi \vdash C \Delta, \Sigma \xrightarrow{s} (\Theta', \Phi \vdash C' \Delta', \Sigma'$$

where:

$$\nu(\Delta', \Theta', \Sigma' \setminus \Delta, \Theta, \Sigma\ .\nu(\vec{n}\ \vec{T})\ .(C' \triangleright n\triangleright t \triangleleft \between C' \equiv \nu(\vec{n}\ \vec{T})\ .(C \triangleright n\triangleright t \triangleleft$$

or symmetrically, swapping the roles of C *and* C *.*

Proof An induction on the derivation of

$$(C \mid C) \equiv \nu(\vec{n})(\vec{T})\,.\,(C \mid n[\mathsf{let}\ x : T = e\ \mathsf{in}\ t])$$

The interesting case is when

$$C \equiv n[\mathsf{let}\ x : T = \mathsf{block}\ \mathsf{in}\ t]$$
$$C \equiv n[\mathsf{let}\ x : T = \mathsf{return}\ v : T\ \mathsf{in}\ t]$$

and

$$n[t\{v/x\}] \mid (C \mid n[\mathsf{let}\ x : T = \mathsf{block}\ \mathsf{in}\ t]) \equiv \nu(\vec{n})(\vec{T})\,.\,(C \mid n[\mathsf{let}\ x : T = e\ \mathsf{in}\ t])$$

so by the induction hypothesis and by induction we have

$$(\Delta, \Phi \vdash C) \quad \Theta, \Sigma \xrightarrow{n[\mathsf{return}\ v]?} (\Delta, \Phi \vdash n[t\{v/x\}]) \quad \Theta, \Sigma$$
$$(\Delta, \Phi \vdash n[t\{v/x\}]) \quad \Theta, \Sigma \xrightarrow{s} (\Delta', \Phi' \vdash \nu(\vec{n})(\vec{T})\,.\,(C' \mid n[\mathsf{let}\ x : T = e\ \mathsf{in}\ t])) \quad \Theta', \Sigma'$$

and

$$(\Delta, \Phi \vdash C) \quad \Theta, \Sigma \xrightarrow{n[\mathsf{return}\ v]} (\Theta, \Phi \vdash n[\mathsf{let}\ x : T = \mathsf{block}\ \mathsf{in}\ t]) \quad \Delta, \Sigma$$
$$(\Theta, \Phi \vdash n[\mathsf{let}\ x : T = \mathsf{block}\ \mathsf{in}\ t]) \quad \Delta, \Sigma \xrightarrow{s} (\Theta', \Phi' \vdash C') \quad \Delta', \Sigma'$$

where

$$\nu(\Delta', \Theta', \Sigma') \setminus \Delta, \Theta, \Sigma\ .\ \nu(\vec{n})(\vec{T})\,.\,(C' \mid n[t] \mid C') \equiv \nu(\vec{n})(\vec{T})\,.\,(C \mid n[t])$$

or symmetrically as required. □

Lemma A. *If* $C \mid C \equiv C$ *and* $C \xrightarrow{\beta} 1$

an so we use the axiom to get

$$(\Delta, \Phi \vdash C : \Theta, \Sigma) \overset{s}{=} (\Delta', \Phi \vdash C' : \Theta', \Sigma')$$

where we define

$$C' \equiv \nu(\vec{n} : \vec{T}, \vec{n}' : \vec{T}') . (C' \mid E \mid n(\text{let } \vec{x} : \vec{T} = \vec{e} \text{ in } t$$

- **Case** $(p \in \mathrm{dom}(C'')$, $n \in \mathrm{dom}(C''$ '

  e ust ave t at

$$C'' \equiv \nu(\vec{p}\ \vec{U}\ .(C''' \mid p\ O\ \mid n(\text{let } y\ U = \text{block in } t\ \mid$$

  Moreover snce C s wr te c ose we ust ave t at t e ax o s

$$p\ O\ \mid n(\text{let } x\ T = p.l(\vec{v}\ \text{ in } t \mid \xrightarrow{\tau}\ p\ O\ \mid n(\text{let } x\ T = O.l(p\ (\vec{v}\ \text{ in } t\mid$$

  n w c case

$$(\Delta, \Phi \mid C\ \Theta, \Sigma \xrightarrow{s\ \nu(\vec{n}\ \vec{T}\ .n(\text{call } p.l(\vec{v}\ \mid} \mid (\Delta, \Phi \mid C'\ \Theta', \vec{n}\ \vec{T}', \Sigma'$$

  w ere we e ne

$$C' \equiv \nu(\vec{n}''\ \vec{T}''\ .(C'' \mid n(\text{let } x\ T = \text{block in } t\ \mid$$

  an we part t on $\{\vec{n}\ \vec{T}\}$ nto $\{\vec{n}'\ \vec{T}', \vec{n}''\ \vec{T}''\}$ suc t at $\{\vec{n}'\} \subseteq \mathrm{fn}(p.l(\vec{v}$ an $\{\vec{n}''\} \cap \mathrm{fn}(p.l(\vec{v}\ = \emptyset$

  e a so ave

$$(\Delta, \Phi \mid C\ \Theta, \Sigma \xrightarrow{s\ \nu(\vec{n}\ \vec{T}\ .n(\text{call } p.l($$

B. Technical preliminaries

In a co ponent ν(Δ . (*p O* □ *C*

A *component for* $\Delta \vdash r \leq s$ trace Θ resp. *for* $\Delta \vdash q \vdash r \leq s$ trace Θ is one of the following

$\nu(\Theta(s \setminus \Theta(q$. $\nu(\mathrm{ref}$ Ref . $\nu(\mathrm{state}_r$ State $\mid \Delta \vdash r^{\sqcup} \leq r$ trace Θ . (
ref val $= \mathrm{state}_r$ $\vdash$
$\prod\{\mathrm{state}_r$ State$(\Delta \vdash r^{\sqcup} \leq s$ trace $\Theta \mid \Delta \vdash r^{\sqcup} \leq r$ trace $\Theta\}$ $\vdash$
$\prod\{p\ l_i = \mathrm{ref.val.inCall}_{p.l_i\ L_i} \mid i = \ldots n\} \mid p \quad l_i \quad L_i \mid i = \ldots n \vdash \Theta, \Theta(s\ \}$ $\vdash$
$\prod\{n \vdash t_n \vdash \mid n$ thread $\vdash \Theta, \Theta(s\ \}$ $\vdash$
$\prod\{n \vdash t_n \vdash \mid n$ thread $\vdash \Delta, \Delta(s$ an $n \vdash$ threads $(q\ \}$

where t_n is a thread at n *for* $\Delta \vdash r \leq s$ trace Θ resp. *for* $\Delta \vdash q \vdash r \leq s$ trace Θ

A *thread at n for* $\Delta \vdash r \leq s$ trace Θ is one of the following

- let x $T = \mathrm{ref.val.out}_T($ in t
  where n is output enable in $\Delta \vdash r$ trace Θ and t is a return $(x$ T thread at n *for* $\Delta \vdash r \leq s$ trace Θ
- let x $T =$ block in t
  where n is input enable in $\Delta \vdash r$ trace Θ and t is a return $(x$ T thread at n *for* $\Delta \vdash r \leq s$ trace Θ

A return $(v$ T *thread at n for* $\Delta \vdash r \leq s$ trace Θ is one of the following

- v
  where n is balance in r
- $\mathrm{ref.val.inReturn}_T(v, t$
  where $r = r\ a\ r \quad a = \nu(\Theta^{\sqcup} . n \vdash \mathrm{call}\, p.l(\vec{v} \vdash \quad n$ is balance in r
  and t is a thread at n *for* $\Delta \vdash r \leq s$ trace Θ
- let y $U =$ return $(v$ T in t
  where $r = r\ a\ r \quad a = \nu(\Theta^{\sqcup} . n \vdash \mathrm{call}\, p.l(\vec{v} \vdash n$ is balance in r
  and t is a return $(y$ U thread at n *for* $\Delta \vdash r \leq s$ trace Θ

Figure — Definition of a component for $\Delta \vdash r \leq s$ trace Θ and for $\Delta \vdash q \vdash r \leq s$ trace Θ

A *thread at n for* $\Delta \vdash q \vdash r \leq s$ trace Θ is one of the following:

- stop
- a thread at n_i or $\Delta \vdash r \leq s$ trace Θ
  where proj n $(q$ = proj n $(r$
- let x $T = p.l(\vec{v}$ in t
  where proj n $(q a$ = proj n $(r$ $a = \nu(\Theta'$ $.n!\text{call}\, p.l(\vec{v}$ $\vdash$ an t is a return $(x$ T
  thread at n_i or $\Delta \vdash r \leq s$ trace Θ
- let x $T =$ return $(v$ U in t
  where proj n $(q a$ = proj n $(r$ $a = \nu(\Theta'$ $.n!\text{return}\, v$ $\vdash$ an t is a return $(x$ T
  thread at n_i or $\Delta \vdash r \leq s$ trace Θ
- let y $U = \text{ref.val.inCall}_{p.l\ L}(\vec{v}$ in let x $T =$ return $(y$ U in t
  where proj n $(q$ = proj n $(r a$ $a = \nu(\Delta'$ $.n!\text{call}\, p.l(\vec{v}$ $\vdash?$ an t is a return $(x$ T
  thread at n_i or $\Delta \vdash r \leq s$ trace Θ
- t
  where proj n $(q$ = proj n $(r a$ $a = \nu(\Delta'$ $.n!\text{return}\, v$ $\vdash?$ an t is a return $(v$ T
  thread at n_i or $\Delta \vdash r \leq s$ trace Θ, i or so be T
- ref.val = new State$(\Delta \vdash r a \leq s$ trace Θ , t
  where proj n $(q$ = proj n $(r a$ an t is a thread at n_i or $\Delta \vdash r a \leq s$ trace Θ
- t
  where $n!t \xrightarrow{\beta} n!t'$ an t' is a thread at n_i or $\Delta \vdash q \vdash r \leq s$ trace Θ

Figure — Definition of a thread $_i$ or $\Delta \vdash q \vdash r \leq s$ trace Θ

Proof An inspection of the definition of Comp ($\Delta \| s$ trace Θ) $\square$

Lemma B. *If* $\Delta \| ra \leq s$ trace Θ *and* $\Delta' \| C$ Θ' *is a component for* $\Delta \| r \leq s$ trace Θ *then* $(\Delta' \| C$ Θ' $\xrightarrow{a}\|$ $(\Delta'' \| C'$ Θ'' *where* C' *is a component for* $\Delta \| ra \leq s$ trace Θ.

Proof By considering the definition of $\Delta \| r$ trace Θ we see that the following cases are exhaustive

- **Case** $a = \nu(\Theta''' .n\langle$return $v\rangle$ an $C \equiv \nu(\Theta''' .C$ ref val $=$ state$_r$ $\| n($let y $U =$ ref.val.out$_U($ in let x $T =$ return$n(y$ U in t

  We have

  $(\Delta' \| C$ Θ'

  $\xrightarrow{\tau}$ $(\Delta' \| \nu(\Theta''' .C$ ref val $=$ state$_r$ $\|$
  $n($let y $U =$ state$_r$.out$_U($ in let x $T =$ return$n(y$ U in $t\rangle$ Θ'

  $\xrightarrow{\beta}\|$ $(\Delta' \| \nu(\Theta''' .C$ ref val $=$ state$_r$ $\|$
  $n\langle$ref.val $=$ new State($\Delta \| ra \leq s$ trace Θ , let y $U = v$ in let x $T =$ return$n(y$ U in $t\rangle$ Θ'

  $\xrightarrow{\tau}$ $(\Delta' \| \nu(\Theta''', \text{state}_{ra}$ State .C ref val $=$ state$_{ra}$ $\|$ state$_{ra}$ State($\Delta \| ra \leq s$ trace Θ $\|$
  $n($let y $U = v$ in let x $T =$ return$n(y$ U in $t\rangle$ Θ'

  $\xrightarrow{\beta}\|$ $(\Delta' \| \nu(\Theta''', \text{state}_{ra}$ State .C ref val $=$ state$_{ra}$ $\|$ state$_{ra}$ State($\Delta \| ra \leq s$ trace Θ $\|$
  $n($let x $T =$ return$n(v$ U in $t\rangle$ Θ'

  $\xrightarrow{a}$ $(\Delta' \| \nu(\text{state}_{ra}$ State .C ref val $=$ state$_{ra}$ $\|$ state$_{ra}$ State($\Delta \| ra \leq s$ trace Θ $\|$
  $n($let x $T =$ block in $t\rangle$ Θ', Θ'''

  which is a component$_i$ or $\Delta \| ra \leq s$ trace Θ as required

- **Case** $a = \nu(\Theta''' .n\langle$call $p.l(\vec{v}$ $\rangle$ an $C \equiv \nu(\Theta''' .C$ ref val $=$ state$_r$ $\| n($let y $U =$ ref.val.out$_U($ in $t\rangle$

  We have

  $(\Delta' \| C$ Θ'

  $\xrightarrow{\tau}$ $(\Delta' \| \nu(\Theta''' .C$ ref val $=$ state$_r$ $\|$
  $n($let y $U =$ state$_r$.out$_U($ in $t\rangle$ Θ'

  $\xrightarrow{\beta}\|$ $(\Delta' \| \nu(\Theta''' .C$ ref val $=$ state$_r$ $\|$
  $n\langle$ref.val $=$ new State($\Delta \| ra \leq s$ trace Θ ,
  let x $T = p.l(\vec{v}$ in ref.val.inReturn$_T(x$, let y $U =$ ref.val.out$_U($ in $t\rangle$ Θ'

  $\xrightarrow{\tau}$ $(\Delta' \| \nu(\Theta''', \text{state}_{ra}$ State .C ref val $=$ state$_{ra}$ $\|$ state$_{ra}$ State($\Delta \| ra \leq s$ trace Θ $\|$
  $n($let x $T = p.l(\vec{v}$ in ref.val.inReturn$_T(x$, let y $U =$ ref.val.out$_U($ in $t\rangle$ Θ'

  $\xrightarrow{a}$ $(\Delta' \| \nu(\text{state}_{ra}$ State .C ref val $=$ state$_{ra}$ $\|$ state$_{ra}$ State($\Delta \| ra \leq s$ trace Θ $\|$
  $n($let x $T =$ block in ref.val.inReturn$_T(x$, let y $U =$ ref.val.out$_U($ in $t\rangle$ Θ', Θ'''

  which is a component$_i$ or $\Delta \| ra \leq s$ trace Θ as required

- **Case** $a = \nu(\Delta''' .n\langle$return $v\rangle$? an $C \equiv C$ ref val $=$ state$_r$ $\| n($let x $T =$ block in ref.val.inReturn$_T(x$, $t\rangle$

e ave

$(\Delta \vdash C \quad \Theta$

$\xrightarrow{a}$ $(\Delta, \Delta' \vdash C \text{ ref val} = \text{state}_r \quad$
$n \triangleright \text{let } x \quad T = v \text{ in ref.val.inReturn}_T(x \; , t \quad \Theta$

$\xrightarrow{\beta}$ $(\Delta, \Delta' \vdash C \text{ ref val} = \text{state}_r \quad$
$n \triangleright \text{ref.val.inReturn}_T(v \; , t \quad \Theta$

$\xrightarrow{\tau}$ $(\Delta, \Delta' \vdash C \text{ ref val} = \text{state}_r \quad$
$n \triangleright \text{state}_r.\text{inReturn}_T(v \; , t \quad \Theta$

$\xrightarrow{\beta}$ $(\Delta, \Delta' \vdash C \text{ ref val} = \text{state}_r \quad$
$n \triangleright \text{ref.val} = \text{new State}(\Delta \vdash ra \leq s \quad \text{trace } \Theta \quad , t \quad \Theta$

$\xrightarrow{\tau}$ $(\Delta, \Delta' \vdash C \; \nu(\text{state}_{ra} \quad \text{State} \quad . \text{ref val} = \text{state}_{ra} \quad \text{state}_{ra} \; \text{State}(\Delta \vdash ra \leq s \quad \text{trace } \Theta \quad$
$n \triangleright t \quad \Theta$

w c s a co ponent $_i$ or $\Delta \vdash ra \leq s$ trace Θ as requre

Case $a = \nu(\Delta' \quad . n \triangleright \text{call } p.l(\vec{v} \; ?$ an $\quad C \equiv C \text{ ref val} = \text{state}_r \quad n \triangleright \text{let } x \quad T = \text{block in } t$

e ave

$(\Delta \vdash C \quad \Theta$

$\xrightarrow{a}$ $(\Delta, \Delta' \vdash C \text{ ref val} = \text{state}_r \quad$
$n \triangleright \text{let } y \quad U = p.l(\vec{v} \quad \text{in let } x \quad T = \text{return } (y \quad U \quad \text{in } t \quad \Theta$

$\xrightarrow{\beta}$ $(\Delta, \Delta' \vdash C \text{ ref val} = \text{state}_r \quad$
$n \triangleright \text{let } y \quad U = \text{ref.val.inCall}_{p.l \; L}(\vec{v} \quad \text{in let } x \quad T = \text{return } (y \quad U \quad \text{in } t \quad \Theta$

$\xrightarrow{\tau}$ $(\Delta, \Delta' \vdash C \text{ ref val} = \text{state}_r \quad$
$n \triangleright \text{let } y \quad U = \text{state}_r.\text{inCall}_{p.l \; L}(\vec{v} \quad \text{in let } x \quad T = \text{return } (y \quad U \quad \text{in } t \quad \Theta$

$\xrightarrow{\beta}$ $(\Delta, \Delta' \vdash C \text{ ref val} = \text{state}_r \quad$
$n \triangleright \text{ref.val} = \text{new State}(\Delta \vdash ra \leq s \quad \text{trace } \Theta \quad ,$
$\text{let } y \quad U = \text{ref.val.out}$

for $\Delta \Box q \Box r \leq s$ trace Θ n F ures an w t e nten e ean n t at a co ponent or $\Delta \Box q \Box r \leq s$ trace Θ as per or e t e trace q an t s s re ate to so e pre x o s ote t at as pre x or er n $\leq$ on traces s conta ne n $\Box$

Case $C \equiv C \text{ ref val} = \text{state}_r \ \| \ n[\text{ref.val} = \text{new State}(\Delta \ \| \ ra \leq s \ \text{trace}\,\Theta \ , t \ \|$

$\xrightarrow{\tau} \nu(\text{state}_{ra}^{T} \ \text{State} \ . C \text{ ref val} = \text{state}_{ra} \ \| \ \text{state}_{ra} \ \text{State}(\Delta \ \| \ ra \leq s \ \text{trace}\,\Theta \ \| \ n[t] \equiv C'$

where t is a trace at n_i or $\Delta \ \| \ ra \leq s \ \text{trace}\,\Theta$

By definition C' is a component or $\Delta \ \| \ q \ \| \ ra \leq s \ \text{trace}\,\Theta$

Case $C \equiv C \ n[\text{let } x \ T = \text{ref.val.out}_T(\ \text{in } t] \xrightarrow{\tau} C \ n[\text{let } x \ T = \text{state}_r.\text{out}_T(\ \text{in } t] \equiv C'$

where $\text{proj}\,n\,(q \ = \text{proj}\,n\,(r \ n$ is output enable in $\Delta \ \| \ r \ \text{trace}\,\Theta$ and t is a return $(x \ T$ trace at n_i or $\Delta \ \| \ r \leq s \ \text{trace}\,\Theta$

If $\Delta \ \| \ ra \leq s \ \text{trace}\,\Theta$ and $a = \nu(\Theta' \ . n[\text{call}\,p.l(\vec{v} \ ]$ then

$$C' \xrightarrow{\beta} \ C \ n[\text{ref.val} = \text{new State}(\Delta \ \| \ ra \leq s \ \text{trace}\,\Theta \ , \ \text{ref.val.inReturn}_U(p.l(\vec{v} \ , \text{let } x \ T = \text{ref.val.out}_T(\ \text{in } t]$$

where C' is a component or $\Delta \ \| \ q \ \| \ r \leq s \ \text{trace}\,\Theta$ as required

If $\Delta \ \| \ ra \leq s \ \text{trace}\,\Theta$ and $a = \nu(\Theta' \ . n[\text{return}\,v]$ then we must have that $r = r \ \nu(\Theta' \ . n[\text{call}\,p.l(\vec{v} \ ]$? r where n is balance in r

- **Case** $(\Delta' \triangleright C\ n[\text{let } x\ T = \text{block in } t] \triangleright \Theta') \xrightarrow{\nu(\Delta''.\ n[\text{call } p.l(\vec{v}\ q')]?} (\Delta', \Delta'' \triangleright C\ n[\text{let } y\ U = p.l(\vec{v}\ \text{ in let } x\ T = \text{return } y\ U \text{ in } t] \triangleright \Theta')$
  where $\text{proj}\, n\,(q) = \text{proj}\, n\,(r)$ n is input enabled in $\Delta \triangleright r$ trace Θ and t is a return $(x\ T)$ trace at n_i or $\Delta \triangleright r \leq s$ trace Θ
  Have
  $$C' \xrightarrow{\beta} C\ n[\text{let } y\ U = \text{ref.val.inCall}_{p.l\ L}(\vec{v}\ \text{ in let } x\ T = \text{return } y\ U \text{ in } t]$$
  with C' as a component for $\Delta \triangleright q[a] \triangleright r \leq s$ trace Θ as required.
- **Case** $(\Delta' \triangleright C\ n[\text{let } x\ T = \text{block in } t] \triangleright \Theta') \xrightarrow{\nu(\Delta''.\ n[\text{return } v]?} (\Delta', \Delta'' \triangleright C\ n[\text{let } x\ T = v \text{ in } t] \triangleright \Theta')$
  where $\text{proj}\, n\,(q) = \text{proj}\, n\,(r)$ n is input enabled in $\Delta \triangleright r$ trace Θ and t is a return $(x\ T)$ trace at n_i or $\Delta \triangleright r \leq s$ trace Θ
  Have
  $$C' \xrightarrow{\beta} C\ n[t\ v/x]$$
  with C' as a component for $\Delta \triangleright q[a] \triangleright r \leq s$ trace Θ as required.
- **Case** $(\Delta' \triangleright \nu(\Theta''.\ C\ n[\text{let } x\ T = p.l(\vec{v}\ \text{ in } t] \triangleright \Theta') \xrightarrow{\nu(\Theta''.\ n[\text{call } p.l(\vec{v}\ q')]} (\Delta' \triangleright C\ n[\text{let } x\ T = \text{block in } t] \triangleright \Theta', \Theta'')$
  where $\text{proj}\, n\,(q[a]) = \text{proj}\, n\,(r)$ and t is a return $(x\ T)$ then at n_i or $\Delta \triangleright r \leq s$ trace Θ
  Have C' as a component for $\Delta \triangleright q[a] \triangleright r \leq s$ trace Θ as required.
- **Case** $(\Delta' \triangleright \nu(\Theta''.\ C\ n[\text{let } x\ T = \text{return } (v\ U \text{ in } t] \triangleright \Theta') \xrightarrow{\nu(\Theta''.\ n[\text{return } v]} (\Delta' \triangleright C\ n[\text{let } x\ T = \text{block in } t] \triangleright \Theta', \Theta'')$
  where $\text{proj}\, n\,(q[a]) = \text{proj}\, n\,(r)$ and t is a return $(x\ T)$ then at n_i or $\Delta \triangleright r \leq s$ trace Θ
  Have C' as a component for $\Delta \triangleright q[a] \triangleright r \leq s$ trace Θ as required. □

We only prove the backward direction now, which follows by induction on Lemma B.1 with Lemma B.2 as the base case and an appropriate use of Corollary B.1.

References

M Aba an L Car e *A Theory Of Objcets* pr n er er a

Abra s y Ja a eesan an r a t Fu t J absn tract oe nn orn

M ner Fu y abstract se ant cs o type λ ca cu *Theoret. Comput. Sci.*

M ner *Communicating and Mobile Systems* Ca br e n vers ty ress

M ner J arrow an D a er A ca cu us o ob e proceses *Inform. and Comput.*

M ner an D an or Barbe b s u at on In *Proc. Int. Colloq. Automata, Languages and Programming* vo u e o *Lecture Notes in Computer Science* pr n er er a

J H Morr s La b a ca cu us o e s o pro ra n an ua es D ssertat on M I

B erce an D an or yp n an subtyp n or ob e processes *Mathematical Structures in Computer Science*

A M tts an I D B tar bservab e propert es o er or er unct ons t at yna ca y create oca na es or at s new? In *Proc. MFCS 93* pa es pr n er er a L C

G ot n LCF cons ere as a pro ra n an ua e *Theoret. Comput. Sci.*