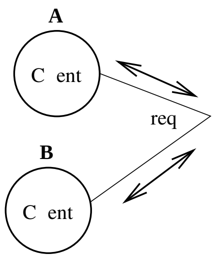
Assigning Types to Processes

NOBUKO YOSHIDA and MATTHEW HENNESSY

ABSTRACT In wide area distributed systems it is now common for *higher-order code*

YPING P OCE E
SS S

(β) and co un cat on (co)—Both these requ re a de n t on of subst tut on of va ues for var ab e


A
C ent
req
B
C ent

on part A −

−o der ve the udge ent t s suf c ent to prove that for any w n $\mathsf{dom}(\Delta \sqcap \Delta)$, $\Gamma \vdash \Delta(w) \leq (\Delta \sqcap \Delta)(w)$ −−here are three poss b t es for w t s e ther n $\mathsf{dom}(\Delta) \cap \mathsf{dom}(\Delta)$, n $\mathsf{dom}(\Delta) - \mathsf{dom}(\Delta)$ or $\mathsf{dom}(\Delta) - \mathsf{dom}(\Delta)$ −In the rst case we have, fro the hypothes s, that $\Gamma \vdash \Delta(w) \leq \Delta_i(w)$ and we ay app y nduct on, on part A to obta n $\Gamma \vdash \Delta(w) \leq \Delta(w) \sqcap \Delta(w)$ and the resu t fo ows, because n th s case $(\Delta \sqcap \Delta)(w) = \Delta(w) \sqcap \Delta(w)$ −

−he other two poss b t es for w are s ar but s p er the nduct ve step s not requ red−

Parts C and D are a so proved s u taneous y, th s t e by s u taneous nduct on on the de n t on of the operators $\sqcap$ and $\sqcup$

(Common)

$$\langle\text{VAR}\rangle\ \frac{\vdash \Gamma, u : \tau, \Gamma' : \mathtt{Env}}{\Gamma, u : \tau, \Gamma' \vdash u : \tau} \qquad \langle\text{CON}\rangle\ \frac{\vdash \Gamma : \mathtt{Env}}{\Gamma \vdash 1 : \mathtt{nat}}\ \text{etc.}$$

$$\langle\text{SUB}_H\rangle\ \frac{\Gamma \vdash P : \rho \quad \Gamma \vdash \rho \leq \rho'}{\Gamma \vdash P : \rho'} \qquad \langle\text{SUB}_N\rangle\ \frac{\Gamma \vdash u : \sigma \quad \Gamma \vdash \sigma \leq \sigma'}{\Gamma \vdash u : \sigma'}$$

(Function)

$$\langle\text{ABS}_H\rangle\ \frac{\Gamma, X : \sigma_H \vdash P : \rho}{\Gamma \vdash \lambda(X : \sigma_H)P : \sigma_H \to \rho} \qquad \langle\text{APP}_H\rangle\ \frac{\Gamma \vdash P : \sigma_H \to \rho \quad \Gamma \vdash Q : \sigma_H}{\Gamma \vdash PQ : \rho}$$

$$\langle\text{ABS}_N\rangle\ \frac{\Gamma, x : \sigma \vdash P : \rho}{\Gamma \vdash \lambda(x : \sigma)P : (x : \sigma) \to \rho} \qquad \langle\text{APP}_N\rangle\ \frac{\Gamma \vdash P : (x : \sigma) \to \rho \quad \Gamma \vdash u : \sigma}{\Gamma \vdash Pu : \rho\{u/x\}}$$

(Process)

$$\langle\text{NIL}\rangle\ \frac{\vdash \Gamma : \mathtt{Env}}{\Gamma \vdash \mathbf{0} : [\,]} \qquad \langle\text{PAR}\rangle\ \frac{\Gamma \vdash P_i : \pi}{\Gamma \vdash P_1 \mid P_2 : \pi} \qquad \langle\text{REP}\rangle\ \frac{\Gamma \vdash P : \pi}{\Gamma \vdash *P : \pi} \qquad \langle\text{RES}\rangle\ \frac{\Gamma, a : \sigma \vdash P : \pi}{\Gamma \vdash (\nu a : \sigma)P : \pi / a}$$

$$\langle\text{OUT}\rangle\ \frac{\pi \vdash_\Gamma u : (\tau_1, \ldots, \tau_n)^{\mathtt{O}} \quad \Gamma \vdash P : \pi \quad \Gamma \vdash V_i : \tau_i \quad \tau_i = \sigma_i \Rightarrow \pi \vdash_\Gamma V_i : \sigma_i}{\Gamma \vdash u\langle V_1, \ldots, V_n\rangle P : \pi}$$

$$\langle\text{IN}\rangle\ \frac{\pi \vdash_\Gamma u : (\tau_1, \ldots, \tau_n)^{\mathtt{I}} \quad \Gamma, x_1 : \tau_1, \ldots, x_n : \tau_n \vdash P : \pi, x_1 : \tau_1, \ldots, x_n : \tau_n}{\Gamma \vdash u(x_1 : \tau_1, \ldots, x_n : \tau_n)P : \pi}$$

FIGURE 3. Typing System for $\lambda\pi_v$

The corresponding elimination, $\langle\text{APP}_N\rangle$ allows dynamic channel instantiation into types during β reduction. If a term P has a type $(x : \sigma) \to \rho$, we can apply a name a whose type is less than σ to P. Then a is substituted for x in ρ.

$$\frac{\Gamma \vdash P : (x : \sigma) \to \rho, \quad \Gamma \vdash a : \sigma}{\Gamma \vdash Pa : \rho\{a/x\}}$$

As an example of the use of this rule consider the channel abstraction $P \equiv \lambda(x : \mathtt{nat})(x\langle\ \rangle \mid b$

s the process type wh ch aps b to the sa e type $(\mathtt{int})^{\mathrm{O}}$ hen w th the output ru e together w th NIL and the abstract on ru es we can estab sh

$$\Delta_{ab} \vdash b\langle\ \rangle \mathbf{0} : [\Delta_b]$$

and therefore

$$\Delta_{ab} \vdash a\langle b\langle\ \rangle \mathbf{0}\rangle \mathbf{0} : [a : \langle\Delta_b\rangle^{\mathrm{O}}]$$

HE INP LE IN he ru e for pre x ng s a stra ghtforward genera sa t on of that n

$$\pi \vdash_{\Gamma} u : (\tau)^{\mathrm{I}} \qquad \Gamma, x : \tau \vdash P : \pi, x : \tau$$

An app cat on of the ru e (O) g ves the udge ent

$$x: (\mathtt{int})^{\mathtt{I}},\ y: (\mathtt{int})^{\mathtt{O}}, z: \mathtt{int} \vdash y\,\langle z\rangle : [\Delta_{xy}]$$

where Δ_{xy} denotes the nterface $\{x: (\mathtt{int})^{\mathtt{I}},\ y: (\mathtt{int})^{\mathtt{O}}\}$. An app cat on of the nput ru e (IN) fo owed by an app cat on of (EP now g ves

$$x: (\mathtt{int})^{\mathtt{I}},\ y: (\mathtt{int})^{\mathtt{O}} \vdash * x\ (z: \mathtt{int})\ y\,\langle z\rangle : [\Delta_{xy}]$$

Now we ay app y the channe abstract on ru e (AB S N tw ce to obta n the fo ow ng type for the forwarder:

$$\vdash \mathsf{Fw} : (x: (\mathtt{int})^{\mathtt{I}}) \to (y: (\mathtt{int})^{\mathtt{O}}) \to [\Delta_{xy}]$$

Let us now see how we can use th s typ ng to ass gn a type to the process R, a so d scussed n the Introduct on:

$$R \Longleftarrow s\,\langle c\rangle\ c\ (y: \tau_{\mathrm{fw}})\ (y\,a\,b)$$

For conven ence τ_{fw} denotes the type ass gned to the forwarder and et us de ne

$$\Delta_R \stackrel{\mathrm{def}}{=} \{a: (\mathtt{int})^{\mathtt{I}}, b: (\mathtt{int})^{\mathtt{O}}, c: ($$

e can now type the co b ned syste —By s proc n F gure , we now

$\Delta \vdash$ [req ((τ

S ub ect educt on aga n t ay be v ewed as a genera sat on of Le a

LEMMA

$$a\ (x \cdot \tau\ ,\dots,x_n \cdot \tau_n)P \xrightarrow{\Gamma,\pi} err \qquad \text{f } \Gamma \not\vdash [a \cdot (\tau\ ,\dots,\tau_n)^{\mathtt{I}}] \leq \pi -$$

$$a\ \langle V\ ,\dots,V_n\rangle P \xrightarrow{\Gamma,\pi} err \qquad \text{f no } \tau_i \text{ s } \tau - \Gamma \vdash [a \cdot (\tau\ ,\dots,\tau_n)^{\mathtt{O}}] \leq \pi \text{ and } \Gamma \vdash V_i \cdot \tau_i -$$

$$\frac{P \xrightarrow{(\Gamma,a\,\sigma),\pi} err}{(\nu a \cdot \sigma)P \xrightarrow{\Gamma,(\pi/a)} err} \qquad \frac{P \xrightarrow{\Gamma,\pi} \text{ or } Q \xrightarrow{\Gamma,\pi}}{P\,|\,Q \xrightarrow{\Gamma,\pi} err} \qquad \frac{P \xrightarrow{\Gamma,\pi} err}{*\,P \xrightarrow{\Gamma,\pi} err}$$

FIG E − un t e errors

Ana ys ng the hypothes s we obta n

$$\begin{array}{ll} \Gamma,x \cdot \sigma \vdash P \cdot [\Delta\ ,x \cdot \sigma] & \text{w th } \Gamma,x \cdot \sigma \vdash [u \cdot (\sigma)^{\mathtt{I}}] \leq [\Delta\ ] \leq [\Delta] \quad x \notin \mathsf{fv}(\Delta\ ) \\ \Gamma \vdash Q \cdot [\Delta\ ] & \text{w th } \Gamma \vdash [u \cdot (\sigma')^{\mathtt{O}}, v \cdot \sigma'] \leq [\Delta\ ] \leq [\Delta] \\ \Gamma \vdash v \cdot \sigma' - & \end{array}$$

Not ng $x \notin \mathsf{fv}(\sigma)$, we can app y Channe narrow ng , Le a −, to obta n $\Gamma \vdash [u \cdot (\sigma)^{\mathtt{I}}] \leq [\Delta\ ]$ − hen we have $\Gamma \vdash \Gamma(u) \leq \Delta(u) \leq \Delta\ (u) \leq (\sigma)^{\mathtt{I}}$ and $\Gamma \vdash \Gamma(u) \leq \Delta(u) \leq \Delta\ (u) \leq (\sigma')^{\mathtt{O}}$, wh ch p y $\Gamma \vdash \sigma' \leq \sigma$−

s ng subsu pt on we then have $\Gamma \vdash v \cdot \sigma$ and so we can app y ubst tut on Le a , Le a −, to obta n $\Gamma \vdash P\{v/x\} \cdot [\Delta\ ,x \cdot \sigma]\{v/x\}$− By ca cu at on th s type s $[\Delta\ ] \sqcup [v \cdot \sigma]$ and we have $\Gamma \vdash [\Delta\ ] \sqcup [v \cdot \sigma] \leq [\Delta\ ] \sqcup [v \cdot \sigma'] \leq [\Delta\ ] \sqcup [\Delta\ ] \leq [\Delta]$− Hence by subsu pt on we have the requ red $\Gamma \vdash P\{v/x\} \cdot [\Delta]$− □

− *ype afety*

Out typ ng syste s an extens on of that for the λ ca cu us fro and that for the π ca cu us fro consequent y t guarantees the absence of the typ ca run t e errors assoc ated w th these anguages− ather than dup cate the for u at on of these nds of errors, wh ch nvo ves the deve op ent co p cated *tagging* notat on, here we concentrate on the nove run t e type errors wh ch our typ ng syste can catch−

Intu t ve y $\Gamma \vdash P \cdot \pi$ should mean that, assu n op

Syntax: others fro F gure

$$\begin{array}{llll} \text{System} & M, N, \ldots & ::= & P \mid N \| M \mid (\nu a : \sigma)N \mid \mathbf{0} \\ \text{Server} & P, Q, \ldots & ::= & \texttt{Spawn}(P) \mid \cdots \quad \text{as in Figure} \end{array}$$

–YPED BEHA IO AL EQ ALI–Y –ypes constra n the behav our of processes and the r env ron ents and consequent y have an pact on when the r behav our shou d be dee ed to be equ va ent– –yped behav oura equ va ences have a ready been nvest gated for var ous process ca cu n papers such as , , , – S ar techn ques cou d be app ed to our anguage, resu t ng n a new typed equ va ence, where equa t es are uenced by the presence of ne gra ned process types– Invest gat on of such equ va ences s an nterest ng research top c, part cu ar y n ts app cat on to the re ne ent of the context equa ty of we eave th s for future wor –

–YPE LIMI–A–ION S One tat on of our typ ng syste s that, wh e na e var ab es n types can be abstracted by channe dependency types

(Free Names)

Terms

$\mathsf{fn}(\mathbf{0}) = \mathsf{fn}(l) = \mathsf{fn}(x) = \emptyset \quad \mathsf{fn}(a) = \{a\}$

$\mathsf{fn}(P \,|\, Q) = \mathsf{fn}(P\,Q) = \mathsf{fn}(P) \cup \mathsf{fn}(Q)$

$\mathsf{fn}(*P) = \mathsf{fn}(P)$

$\mathsf{fn}(u\ \ (x\ \tau\ \ ,\ldots,x_n\ \tau_n)P)$
$= \mathsf{fn}(u) \cup \mathsf{fn}(\tau\ \ ) \cup \ldots \cup \mathsf{fn}(P$

Giacalone, A., Mishra, P. and Prasad, S., Operational and Algebraic Semantics for Facile: A Symmetric Integration of Concurrent and Functional Programming.