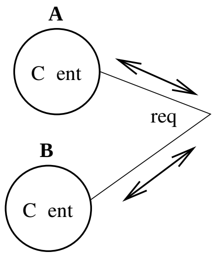
Assigning Types to Processes

NOBUKO YOSHIDA and MATTHEW HENNESSY

ABSTRACT. In w de area d str buted syste s t s now co on 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)

$$(\text{AL})\ \frac{\vdash \Gamma, u : \tau, \Gamma' : \mathtt{Env}}{\Gamma, u : \tau, \Gamma' \vdash u : \tau} \qquad (\text{CON})\ \frac{\vdash \Gamma : \mathtt{Env}}{\Gamma \vdash \ : \mathtt{nat}} \quad \text{etc.}$$

$$(\text{S B}_H)\ \frac{\Gamma \vdash P : \rho \quad \Gamma \vdash \rho \leq \rho'}{\Gamma \vdash P : \rho'} \qquad (\text{S B}_N)\ \frac{\Gamma \vdash u : \sigma \quad \Gamma \vdash \sigma \leq \sigma'}{\Gamma \vdash u : \sigma'}$$

(Function)

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

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

(Process)

$$(\text{NIL})\ \frac{\vdash \Gamma : \mathtt{Env}}{\Gamma \vdash \mathbf{0} : [\ ]} \qquad (\text{PA})\ \frac{\Gamma \vdash P_{\ } : \pi}{\Gamma \vdash P\ | P : \pi} \qquad (\text{EP})\ \frac{\Gamma \vdash P : \pi}{\Gamma \vdash *P : \pi} \qquad (\text{ES})\ \frac{\Gamma, a : \sigma \vdash P : \pi}{\Gamma \vdash (\nu a : \sigma)P : \pi/a}$$

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

$$(\text{IN})\ \frac{\pi \vdash_\Gamma u : (\tau\ , \ldots, \tau_n)^{\mathtt{I}} \qquad \Gamma, x\ : \tau\ , \ldots, x_n : \tau_n \vdash P : \pi, x\ : \tau\ , \ldots, x_n : \tau_n}{\Gamma \vdash u\ (x\ : \tau\ , \ldots, x_n : \tau_n)P : \pi}$$

FIG E .-yp ng S yste for $\lambda\pi_v$

.he correspond ng e nat on (APP$_N$) a ows dyna c channe nstant at on nto types dur ng β reduct on. If a ter P has a type $(x : \sigma) \to \rho$, we can ap p y a na e a whose type s ess than σ to P. .hen a s subst tuted for x n ρ.

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

As an exa p e of the use of th s ru e cons der the channe abstract on $P \equiv \lambda(x : \mathtt{nat})(x\ \langle\ \rangle\,|\,b$

s the process type wh ch aps b to the sa e type $(\mathtt{int})^{\mathtt{o}}$ 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

$$\Delta_{ab} \vdash a\langle b\langle \ \rangle \mathbf{0}\rangle \mathbf{0} \, [a \langle \Delta_b \rangle^{\mathtt{o}}]$$

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

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

An application of the rule (OUT) gives the judgement

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

where Δ_{xy} denotes the interface $\{x : (\mathtt{int})^{\mathtt{I}},\ y : (\mathtt{int})^{\mathtt{O}}\}$. An application of the input rule (IN), followed by an application of (REP) now gives

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

Now we may apply the channel abstraction rule (ABS$_N$) twice to obtain the following type for the forwarder:

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

Let us now see how we can use this typing to assign a type to the process R, also discussed in the Introduction:

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

For convenience τ_{fw} denotes the type assigned to the forwarder and let us define

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

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

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

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

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

FIG E – un t e errors

Ana ys ng the hypothes s we obta n

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

Not ng $x \notin \mathsf{fv}(\sigma)$, we can app y Channe narrow ng , Le a –, to obta n $\Gamma \vdash [u : (\sigma)^{\mathtt{I}}] \leq [\Delta\ ]$– 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 $\Gamma \vdash v : \sigma$ and so we can app y ubst tut on Le a , Le a –, to obta n $\Gamma \vdash P\{v/x\} : [\Delta\ ,x : \sigma]\{v/x\}$–By ca cu at on th s type s $[\Delta\ ] \sqcup [v : \sigma]$ and we have $\Gamma \vdash [\Delta\ ] \sqcup [v : \sigma] \leq [\Delta\ ] \sqcup [v : \sigma'] \leq [\Delta\ ] \sqcup [\Delta\ ] \leq [\Delta]$–Hence by subsu pt on we have the requ red $\Gamma \vdash P\{v/x\} : [\Delta]$– □

– 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 $\Gamma \vdash P : \pi$ shou d ean that, assu ng the env ron ent Γ, the process P sat s es the *interface* π–If π s the und fferent ated type `proc` then, v ewed as an nterface, t prov des no nfor at on–However f t has the for $[\Delta]$ th s eans that P can use *at most* the resources ent oned n Δ oreover these resources can on y be used accord ng to the capab t es they are ass gned n Δ– A s p e for a sat on of th s ntu t ve dea s g ven n F gure , us ng a unary pred cate $P \xrightarrow{\Gamma,\pi}_{err}$– he rst two c auses are the ost s gn cant– he rst says that, re at ve to Γ, P v o ates the nterface π f t can nput on the channe a but the nterface π does not ass gn any nput capab ty to a the second s s ar not a unaon

Syntax: others fro F gure –

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

–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 , , , – 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 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

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

$\mathsf{fn}(P \mid Q) = \mathsf{fn}(PQ) = \mathsf{fn}(P) \cup \mathsf{fn}(Q)$

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

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

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