Continuations, mobile processes, all the things...

Julien Gabet

Mars-June, 2018

Memo/garbage part

$M, N ::= x; \lambda x.M; MN$
$(\lambda x.M)N \to_\beta M[N/x]$ terms tend to get bigger

If $C[\,]$ is a context, and $M \to_\beta N$
then $C[M] \to_\beta C[N]$

$P, Q ::= u(xy).P; \bar{u}xy.P; P|Q; (\nu x)P|!P$
$u(xy).P|\bar{u}ab.Q \to P[a/x, b/y]|Q$

If $P \to Q$ then $C[P] \to C[Q]$ (with necessary hypothesis on context C)
If $P \equiv P' \to Q \equiv Q'$ then $P \to Q$

Krivine Abstract Machine (KAM)
$M \star \Pi \star \mathcal{E}$

$$\begin{aligned} MN \star \Pi \star \mathcal{E} &\to M \star (N, \mathcal{E}).\Pi \star \mathcal{E} \\ \lambda x.M \star (N, \mathcal{E}).\Pi \star \mathcal{F} &\to M \star \Pi \star \mathcal{F}, s \mapsto (N, \mathcal{E}) \\ x \star \Pi \star \mathcal{E}, x \mapsto (M, \mathcal{F}) &\to M \star \Pi \star \mathcal{F} \end{aligned}$$

For exponentials :
  $!P \simeq !P|!P$
  $(\nu u)!u(x).P \simeq 0$
idea : $!P|Q \simeq !P|!P|Q \ \ \forall Q$

$$\begin{aligned} [\![(M, \mathcal{E}).\Pi]\!]_u &= (\nu m)(\nu v)(\bar{u}mv|!m(x)[\![M, \mathcal{E}]\!]_x|[\![\Pi]\!]_v) \\ [\![M, (x_i \mapsto (M_i, \mathcal{E}_i))_{i=1..k}]\!]_u &= (\nu x_1) \cdots (\nu x_k)([\![M]\!]_u|!x_1(u).[\![M_1, \mathcal{E}_1]\!]_u| \cdots) \\ [\![MN]\!]_u &= (\nu v)(\nu n)([\![M]\!]_v|\bar{v}nu|!n(x).[\![N]\!]_x) \\ [\![\lambda x.M]\!]_u &= u(xv).[\![M]\!]_v \\ [\![x]\!]_u &= \bar{x}u \end{aligned}$$

We want $M \star \Pi \star \mathcal{E} \to M' \star \Pi' \star \mathcal{E}'$ iff $[\![M, \mathcal{E}]\!]_u|[\![\Pi]\!]_u \to [\![M', \mathcal{E}']\!]_v|[\![\Pi']\!]_v$

- equiv $\simeq$ bisimulation
- the traduction goes well

Definition

A binary relation S is a reduction bisimulation if, forall $(P, Q) \in S$

(1) $P \xrightarrow{\tau} P'$ implies $Q \xrightarrow{\tau} Q'$ for some Q' with $(P', Q') \in S$
(2) $Q \xrightarrow{\tau} Q'$ implies $P \xrightarrow{\tau} P'$ for some P' with $(P', Q') \in S$

Definition (*Observability :*)

$P \downarrow_x$ if P can make an input action of subject x
$P \downarrow_{\bar{x}}$ if P can make an output action of subject x.

Definition (*Image-finite process :*)

P is image-finite if, for all derivative Q of P and any action $\alpha, \exists n \geq 0$ and $Q_1, \cdots Q_n$ such that $Q \stackrel{\alpha}{\Rightarrow} Q'$ implies $Q' = Q_i$ for some i.
where $\Rightarrow$ is the reflexive transitive closure of $\xrightarrow{\tau}$ and $\stackrel{\alpha}{\Rightarrow}$ is $\Rightarrow\xrightarrow{\alpha}\Rightarrow$ for some action α.

Rules for base-π

Value-typing

$$\frac{}{\Gamma \vdash basval : B}\ \text{TV-BASVAL} \qquad \frac{}{\Gamma, x : T \vdash x : T}\ \text{TV-NAME}$$

Process typing

$$\frac{\Gamma \vdash P : \diamondsuit \quad \Gamma \vdash Q : \diamondsuit}{\Gamma \vdash P|Q : \diamondsuit}\ \text{T-PAR} \qquad \frac{\Gamma \vdash P : \diamondsuit \quad \Gamma \vdash Q : \diamondsuit}{\Gamma \vdash P + Q : \diamondsuit}\ \text{T-SUM}$$

$$\frac{\Gamma \vdash v : \sharp T \quad \Gamma \vdash w : \sharp T \quad \Gamma \vdash P : \diamondsuit}{\Gamma \vdash [v = w]P : \diamondsuit}\ \text{T-MAT} \qquad \frac{}{\Gamma \vdash 0 : \diamondsuit}\ \text{T-NIL}$$

$$\frac{\Gamma \vdash P : \diamondsuit}{\Gamma \vdash !P : \diamondsuit}\ \text{T-REP} \qquad \frac{\Gamma, x : L \vdash P : \diamondsuit}{\Gamma \vdash (\nu x : L)P : \diamondsuit}\ \text{T-RES} \qquad \frac{\Gamma \vdash P : \diamondsuit}{\Gamma \vdash \tau.P : \diamondsuit}\ \text{T-TAU}$$

$$\frac{\Gamma \vdash v : \sharp T \quad \Gamma, x : T \vdash P : \diamondsuit}{\Gamma \vdash v(x).P : \diamondsuit}\ \text{T-INP} \qquad \frac{\Gamma \vdash v : \sharp T \quad \Gamma \vdash w : T \quad \Gamma \vdash P : \diamondsuit}{\Gamma \vdash \bar{v}w.P : \diamondsuit}\ \text{T-OUT}$$

Types : $S, T ::= V$ value type
$|L$ link type
$|\diamond$ behaviour type

Value types : $V ::= B$ basic type
Link types : $L ::= \sharp V$ connexion type
Environments : $\Gamma ::= \Gamma, x : L | \Gamma, x : V | \emptyset$

Transitions for base-π

$$\frac{}{\bar{a}w.P \xrightarrow{\bar{a}w} P}\ \text{OUT} \qquad \frac{}{a(x).P \xrightarrow{aw} P\{w/x\}}\ \text{INP} \qquad \frac{}{\tau P \xrightarrow{\tau} P}\ \text{TAU}$$

$$\frac{P \xrightarrow{\alpha} P'}{[x = x]P \xrightarrow{\alpha} P'}\ \text{MAT} \qquad \frac{P \xrightarrow{\alpha} P'}{P + Q \xrightarrow{\alpha} P'}\ \text{SUM-L} \qquad \frac{P \xrightarrow{\alpha} P'}{P|Q \xrightarrow{\alpha} P'|Q}\ \text{PAR-L}\ (bn(\alpha) \cap fn(Q) = \emptyset)$$

$$\frac{P \xrightarrow{(\nu \widetilde{z} : \widetilde{T})\bar{a}v} P' \quad Q \xrightarrow{av} Q'}{P|Q \xrightarrow{\tau} (\nu \widetilde{z} : \widetilde{T})(P'|Q')}\ \text{COMM-L}\ (\widetilde{z} \cap fn(Q) = \emptyset)$$

$$\frac{P \xrightarrow{\alpha} P'}{(\nu x : T)P \xrightarrow{\alpha} (\nu x : T)P'}\ \text{RES}\ (x \notin n(\alpha)) \qquad \frac{P \xrightarrow{\alpha} P'}{!P \xrightarrow{\alpha} P'|!P}\ \text{REP-ACT}$$

$$\frac{P \xrightarrow{(\nu \widetilde{z} : \widetilde{T})\bar{a}v} P'}{(\nu x : T)P \xrightarrow{(\nu \widetilde{z} : \widetilde{T}, x : T)} P'}\ \text{OPEN}\ (x \in fn(v), x \notin \{\widetilde{z}, a\})$$

$$\frac{P \xrightarrow{(\nu \widetilde{z} : \widetilde{T})\bar{a}v} P' \quad P \xrightarrow{av} P''}{!P \xrightarrow{\tau} (\nu \widetilde{z} : \widetilde{T})(P'|P'')|!P}\ \text{REP-COMM}\ (\widetilde{z} \cap fn(P) = \emptyset)$$

$$\frac{\text{Si } v \text{ n'est pas un nom}}{\bar{v}w.P \xrightarrow{\tau} wrong}\ \text{OUTERR} \qquad \frac{\text{Si } v \text{ n'est pas un nom}}{v(x).P \xrightarrow{\tau} wrong}\ \text{INPERR}$$

$$\frac{\text{Si } v \text{ ou } w \text{ n'est pas un nom}}{[v = w]P \xrightarrow{\tau} wrong}\ \text{MATERR}$$

simply-typed π-calculus : same but with value types $V ::= B$ basic type
$|L$ link type

i/o types

Grammar : same + $L ::= iV|oV$ (input and output capabilities)
Subtyping rules

SUB-REFL SUB-TRANS

SUB-♯I SUB-♯O

SUB-II SUB-OO

SUB-BS

Typing rules
T-INPS replaces T-INP
T-OUTS replaces T-OUT
SUBSUMPTION

Linear types

fill this in later