Communication Interference in Mobile Boxed Ambients

FST&TCS 2002 (to appear)

joint with: M. Bugliesi, S. Crafa, M. Merro

Vladimiro Sassone

University of Sussex,
Mobile Ambients

Both administrative domains and computational environments (Cardelli-Gordon)

- Subjective movements

\[
\begin{align*}
    n[\text{in } m.P \mid Q] \mid m[R] &\longrightarrow m[n[P \mid Q] \mid R] \\
    m[n[\text{out } m.P \mid Q] \mid R] &\longrightarrow n[P \mid Q] \mid m[R]
\end{align*}
\]

- Process interaction

\[
\begin{align*}
    n[\langle M \rangle.P \mid (x).Q] &\longrightarrow n[P \mid Q\{x := M\}],
\end{align*}
\]

- Boundary dissolver

\[
\begin{align*}
    \text{open } n.P \mid n[Q] &\longrightarrow P \mid Q.
\end{align*}
\]
Interferences in Mobile Ambients

The inherent nondeterminism of movement may go wild: Grave Interferences.

\[ k[n \text{ in } m.P \mid \text{ out } k.R] \mid m[Q] \]
Interferences in Mobile Ambients

- The inherent nondeterminism of movement may go wild: Grave Interferences.

\[ k[n[ \text{in} \ m \cdot P \ | \ \text{out} \ k \cdot R ] \ | \ m[ Q ] ] \]

- Introducing Safe Ambients (Levi-Sangiorgi)

\[ n[ \text{in} \ m \cdot P \ | \ Q ] \ | \ m[ \text{in} \ m \cdot R \ | \ S ] \rightarrow m[ n \cdot P \ | \ Q ] \ | \ R \ | \ S ] \]

- Co-capabilities and single-threadedness rule out grave interferences
Interferences in Mobile Ambients

- The inherent nondeterminism of movement may go wild: **Grave Interferences**.

\[ k[n[\text{in } m.P \ | \ \text{out } k.R] \ | \ m[Q]] \]

- Introducing **Safe Ambients** (Levi-Sangiorgi)

\[ n[\text{in } m.P \ | \ Q] \ | \ m[\overline{\text{in } m.R} \ | \ S] \rightarrow m[n[P \ | \ Q] \ | \ R \ | \ S] \]

- **Co-capabilities and single-threadedness** rule out grave interferences

- Safe Ambients with passwords have a conveniently treatable semantics. (Merro-Hennessy)

\[ n[\text{in } (m,k).P \ | \ Q] \ | \ m[\overline{\text{in } (m,k).R} \ | \ S] \rightarrow m[n[P \ | \ Q] \ | \ R \ | \ S] \]
Mobile Boxed Ambients

.open’s nature of ambient dissolver is a potential source of problems.

Direct communication as alternative source of expressiveness: Mobile Boxed Ambients (Bugliesi et al.). Perform I/O on a subambient \( n \)'s local channel (viz. \( (x)^n \)) as well as from the parent’s local channel (viz. \( (x)^\uparrow \))

\[
(x)^n.P \mid \mathcal{N}[M].Q \mid R \rightarrow P\{x := M\} \mid \mathcal{N}[Q \mid R]
\]

\[
\langle M \rangle.P \mid \mathcal{N}[(x)^\uparrow].Q \mid R \rightarrow P \mid \mathcal{N}[Q\{x := M\} \mid R].
\]
Mobile Boxed Ambients

**open**’s nature of ambient dissolver is a potential source of problems.

Direct communication as alternative source of expressiveness: Mobile Boxed Ambients (Bugliesi et al.). Perform I/O on a subambient $n$’s local channel (viz. $(x)^n$) as well as from the parent’s local channel (viz. $(x)^\uparrow$)

$$
(x)^n . P | n[\langle M \rangle . Q | R] \rightarrow P\{x := M\} | n[Q | R] \\
\langle M \rangle . P | n[(x)^\uparrow . Q | R] \rightarrow P | n[Q\{x := M\} | R].
$$

But it is a great source of non-local nondeterminism and communication interference.

$$
m[(x)^n . P | n[\langle M \rangle | (x).Q | k[(x)^\uparrow . R])]]
$$
Mobile Boxed Ambients

- open’s nature of ambient dissolver is a potential source of problems.

Direct communication as alternative source of expressiveness: Mobile Boxed Ambients (Bugliesi et al.). Perform I/O on a subambient n’s local channel (viz. \((x)^n\)) as well as from the parent’s local channel (viz. \((x)^\uparrow\))

\[
(x)^n.P \mid n[\langle M \rangle.Q \mid R] \rightarrow P\{x := M\} \mid n[Q \mid R]
\]

\[
\langle M \rangle.P \mid n[(x)^\uparrow.Q \mid R] \rightarrow P \mid n[Q\{x := M\} \mid R].
\]

- But it is a great source of non-local nondeterminism and communication interference.

\[
m[(x)^n.P \mid n[\langle M \rangle \mid (x).Q \mid k[(x)^\uparrow.R]]]
\]
Introducing NBA: Communication

NBA: a fresh foundation based on: each ambient comes equipped with two mutually non-interfering channels, for \textit{local} and \textit{upward} communications.

\[(x)^n . P \mid n[\langle M \rangle^\hat{x} . Q \mid R] \longrightarrow P\{x := M\} \mid n[ Q \mid R] \]

\[\langle M \rangle^n . P \mid n[(x)^\hat{x} . Q \mid R] \longrightarrow P \mid n[ Q\{x := M\} \mid R] \]
Introducing NBA: Communication

NBA: a fresh foundation based on: each ambient comes equipped with two mutually non-interfering channels, for local and upward communications.

\[(x)^n P \mid n[(M)^\wedge] Q \mid R \rightarrow P\{x := M\} \mid n[Q \mid R]\]

\[\langle M\rangle^n P \mid n[(x)^\wedge] Q \mid R \rightarrow P \mid n[Q\{x := M\} \mid R]\]
Introducing NBA: Communication

**NBA**: a fresh foundation based on: each ambient comes equipped with two mutually non-interfering channels, for **local** and **upward** communications.

\[
(x)^n \cdot P | n[\langle M \rangle \cdot Q | R] \longrightarrow P\{x := M\} | n[Q | R]
\]

\[
\langle M \rangle^n \cdot P | n[(x) \cdot Q | R] \longrightarrow P | n[Q\{x := M\} | R]
\]

- Good algebraic laws; simple type system;
- Expressiveness??
Introducing NBA: Communication

NBA: a fresh foundation based on: each ambient comes equipped with two mutually non-interfering channels, for local and upward communications.

\[(x)^n.P \mid n[(M)^\hat{x}.Q \mid R] \rightarrow P\{x := M\} \mid n[Q \mid R]\]

\[(M)^n.P \mid n[(x)^\hat{x}.Q \mid R] \rightarrow P \mid n[Q\{x := M\} \mid R]\]

- Good algebraic laws; simple type system;
- Expressiveness??
- Hmm, rather poor: \( n[P] \) cannot, for instance, communicate with children it doesn’t know statically. It can never learn about incoming ambients, and will never be able to talk to them.
Introducing NBA: Mobility

Essentially, our idea is to introduce co-actions of the form $\text{enter}(x)$ which have the effect of binding the variable $x$.

Such a purely binding mechanism does not provide a way control of access, but only to registers it. As a (realistic) access protocol where newly arrived agents must register themselves to be granted access to local resources.
Essentially, our idea is to introduce co-actions of the form $\text{enter}(x)$ which have the effect of binding the variable $x$.

Such a purely binding mechanism does not provide a way control of access, but only to registers it. As a (realistic) access protocol where newly arrived agents must register themselves to be granted access to local resources.

Need a finer mechanism of access control:

$$
a[\text{enter}(b, k).P_1 | P_2] \mid b[\text{enter}(x, k).Q_1 | Q_2] \rightarrow b[a[P_1 | P_2] | Q_1\{x := a\} | Q_2]
$$

This represents an access protocol where the credentials of incoming processes ($k$ in the rule above) are controlled, as a preliminary step to the registration protocol.
Introducing NBA: Mobility

Essentially, our idea is to introduce co-actions of the form \( \text{enter}(x) \) which have the effect of binding the variable \( x \).

Such a purely binding mechanism does not provide a way control of access, but only to registers it. As a (realistic) access protocol where newly arrived agents must register themselves to be granted access to local resources.

Need a finer mechanism of access control:

\[
\begin{align*}
& a[ \text{enter}(b,k).P_1 | P_2 ] | b[ \text{enter}(x,k).Q_1 | Q_2 ] \rightarrow b[ a[P_1 | P_2] | Q_1 \{ x := a \} | Q_2 ]
\end{align*}
\]

This represents an access protocol where the credentials of incoming processes (\( k \) in the rule above) are controlled, as a preliminary step to the registration protocol.
## NBA: Syntax

### Names: $a, b, \ldots n, x, y, \ldots \in \mathbb{N}$

### Locations:

- $\eta ::= a$ nested names
- $\hat{\cdot}$ enclosing ambient
- $\star$ local

### Messages:

- $M, N ::= a$ name
- $\text{enter}(M, N)$ may enter
- $\text{exit}(M, N)$ may exit
- $M.N$ path

### Processes:

- $P ::= \mathbf{0}$ nil process
- $P_1 | P_2$ composition
- $(\nu n)P$ restriction
- $!\pi.P$ replication
- $M[P]$ ambient
- $\pi.P$ prefixing

### Prefixes:

- $\pi ::= M$ messages
- $(x_1, \ldots, x_k)\eta$ input
- $\langle M_1, \ldots, M_k \rangle\eta$ output
- $\text{enter}(x, M)$ allow enter
- $\text{exit}(x, M)$ allow exit
NBA: Reduction Semantics

mobility
\[ n[\text{enter}(m, k) . P_1 | P_2] \mid m[\text{enter}(x, k) . Q_1 | Q_2] \rightarrow m[n[P_1 | P_2] | Q_1\{x := n\} | Q_2] \]
\[ n[m[\text{exit}(n, k) . P_1 | P_2] | Q] \mid \text{exit}(x, k).R \rightarrow m[P_1 | P_2] \mid n[Q] | R\{x := m\} \]

communication
\[ (\tilde{x}).P \mid \langle \tilde{M} \rangle .Q \rightarrow P\{\tilde{x} := \tilde{M}\} | Q \]
\[ (\tilde{x})^n.P \mid n[\langle \tilde{M} \rangle \hat{\}.Q \mid R] \rightarrow P\{\tilde{x} := \tilde{M}\} \mid n[Q \mid R] \]
\[ \langle \tilde{M} \rangle .P \mid n[(\tilde{x})^n .Q \mid R] \rightarrow P \mid n[Q\{\tilde{x} := \tilde{M}\} \mid R] \]

structural congruence
\[ P \equiv Q \quad Q \rightarrow R \quad R \equiv S \text{ implies } P \rightarrow S \]
NBA: Behavioural Equivalence

Barbs

\[ P \downarrow_n \text{ iff } P \equiv (\nu \bar{m})(n[\text{enter}(x, k).Q \mid R] \mid S), \text{ for } \{n, k\} \cap \{\bar{m}\} = \emptyset. \]

\[ P \vdash_n \text{ iff } P \implies P' \text{ and } P' \downarrow_n. \]
NBA: Behavioural Equivalence

Barbs

\[ P \downarrow_n \text{ iff } P \equiv (\nu \bar{m})(n[\text{enter}(x,k).Q \mid R] \mid S), \quad \text{for } \{n,k\} \cap \{\bar{m}\} = \emptyset. \]

\[ P \Downarrow_n \text{ iff } P \rightarrow P' \text{ and } P' \downarrow_n. \]

A relation \( R \) is reduction closed if

\[ P R Q \text{ and } P \rightarrow P' \text{ implies } Q \Rightarrow Q' \text{ with } P' R Q'; \]

it is barb preserving if \( P R Q \) and \( P \downarrow_n \) implies \( Q \Downarrow_n \).

Reduction barbed congruence, written \( \cong \), is the largest congruence relation over processes which is reduction closed and barb preserving.

Note: We could equivalently observe \( (\cdot)^\wedge \).
The rest of the talk

- Two small examples
- A few equational laws
- LTS characterization of reduction barbed bisimulation congruence.
- A type system
- An encoding of BA into NBA: \( BA \subseteq NBA + \text{Guarded Choice} \)
A one-to-one communication server

Let \( w(k) \) be a bidirectional forwarder for any pair of incoming ambients.

\[
\begin{align*}
    w(k) & \triangleq w[ \overline{\text{enter}(x, k).\text{enter}(y, k).(!z)^x.(z)^y | !(z)^y.(z)^x) ] \\
\end{align*}
\]

An agent can be defined as: \( A(a, k, P, Q) \triangleq a[\text{enter}(w, k).P | \text{exit}(w, k).Q] \) and a communication server as:

\[
\begin{align*}
    o2o(k) & = (\nu r)(r[\langle \rangle^\hat{} ] | !(\) r. (w(k) | \overline{\text{exit}(\_, k).\text{exit}(\_, k).r[\langle \rangle^\hat{} ])) \\
\end{align*}
\]
A one-to-one communication server

Let \( \text{w}(k) \) be a bidirectional forwarder for any pair of incoming ambients.

\[
\text{w}(k) \triangleq \text{w}[ \overline{\text{enter}}(x, k).\overline{\text{enter}}(y, k).!(z)^x.(z)^y | !(z)^y.(z)^x ) ]
\]

An agent can be defined as: \( \text{A}(a, k, P, Q) \triangleq a[\text{enter}\langle w, k \rangle.P | \text{exit}\langle w, k \rangle.Q] \) and a communication server as:

\[
\circ\circ\circ(k) = (\nu r) ( r[\langle \rangle^\hat{\rangle} ] | !(\langle \rangle^\hat{\rangle}.(w(k) | \overline{\text{exit}}(-, k).\overline{\text{exit}}(-, k).r[\langle \rangle^\hat{\rangle}]) )
\]

It can be proved that:

\[
(\nu k)(\circ\circ\circ(k) | \text{A}(k, a_1, \langle M \rangle^\hat{\rangle}.P_1, Q_1) | \text{A}(k, a_2, (x)^\hat{\rangle}.P_2\{x\}, Q_2) | \Pi_{i \in I} \text{A}(K, a_i, R_i, S_i) )
\]

\[
\implies \cong (\nu k)(\circ\circ\circ(k) | a_1[P_1 | Q_1] | a_2[P_1\{x := M\} | Q_2] | \Pi_{i \in I} \text{A}(K, a_i, R_i, S_i) )
\]

that is, once two agents engage in communication no other agent knowing the key \( k \) can interfere with their completing the exchange.
A print server

The following process assigns a progressive number to incoming jobs.

\[
\text{enqueue}_k \triangleq (\forall c) (c[1^\uparrow] | !(n)^c.\text{enter}(x, k).(n)^x.c[n + 1^\uparrow])
\]
A print server

The following process assigns a progressive number to incoming jobs.

\[
\text{enqueue}_k \triangleq (\nu c) \ (c[\langle 1 \rangle \hat{\rangle}] \mid !(n)^c.\text{enter}(x, k).\langle n \rangle^x. c[\langle n + 1 \rangle \hat{\rangle}])
\]

We can turn it into a print server (which consumes such numbers).

\[
\text{prtsrv}(k) \triangleq k[\text{enqueue}_k | \text{print}]
\]

\[
\text{print} \triangleq (\nu c) \ (c[\langle 1 \rangle \hat{\rangle}] \mid !(n)^c.\text{exit}(x, n).\langle \text{data} \rangle^x. (P\{\text{data}\} | c[\langle n + 1 \rangle \hat{\rangle}])
\]

A client then acts as:

\[
\text{job}(M, k) \triangleq (\nu p)p[\text{enter}(k, k).\langle n \rangle \hat{\rangle}. (\nu q)q[\text{exit}(p, n).\langle M \rangle \hat{\rangle}]
\]

It enters the server \text{prtsrv}(k) (using \text{enqueue}), it is assigned a number that it uses as a password to carry job \(M\) to \text{print} (which eventually will bind it to \text{data} in \(P\). (Dynamic name discovery and passwords are fundamental here.)
Some Equational Laws

Garbage Collection laws

\[ l[ (\tilde{x}_i)^n.P \mid (\tilde{x}).Q \mid \langle \tilde{M} \rangle^m.R ] \simeq 0 \]

\[ l[ (\tilde{x})^n.P \mid \langle \tilde{M} \rangle.P \mid \langle \tilde{M} \rangle^m.P ] \simeq 0 \]
Some Equational Laws

Garbage Collection laws

- \( l[ (\tilde{x}_i)^n.P \mid (\tilde{x}).Q \mid \langle \tilde{M} \rangle^m.R ] \equiv 0 \)
- \( l[ (\tilde{x})^n.P \mid \langle \tilde{M} \rangle.P \mid \langle \tilde{M} \rangle^m.P ] \equiv 0 \)

Communication laws

- \( l[ \langle \tilde{M}_0 \rangle^\wedge \mid \langle \tilde{M}_1 \rangle^\wedge ] \equiv l[\langle \tilde{M}_0 \rangle^\wedge ] \mid l[\langle \tilde{M}_1 \rangle^\wedge ] \)
- \( l[(\tilde{x}).P \mid \langle \tilde{M} \rangle.Q] \equiv l[P\{\tilde{x} := \tilde{M}\} \mid Q] \)
- \( (\nu l)((\tilde{x})^l.P \mid l[\langle \tilde{M} \rangle^\wedge.Q]) \equiv (\nu l)(P\{\tilde{x} := \tilde{M}\} \mid l[Q]) \)
- \( m[(\tilde{x})^l.P \mid l[\langle \tilde{M} \rangle^\wedge.Q]] \equiv m[P\{\tilde{x} := \tilde{M}\} \mid l[Q]]\)
Some Equational Laws

Garbage Collection laws

\[ l[(\bar{x}_i^n \cdot P | (\bar{x}).Q | \langle \bar{M} \rangle^m \cdot R)] \cong 0 \]

\[ l[(\bar{x}^n \cdot P | \langle \bar{M} \rangle \cdot P | \langle \bar{M} \rangle^m \cdot P)] \cong 0 \]

Communication laws

\[ l[\langle \bar{M}_0 \rangle ^\wedge | \langle \bar{M}_1 \rangle ^\wedge] \cong l[\langle \bar{M}_0 \rangle ^\wedge] | l[\langle \bar{M}_1 \rangle ^\wedge] \]

\[ l[(\bar{x}).P | \langle \bar{M} \rangle \cdot Q] \cong l[P\{\bar{x} := \bar{M}\} | Q] \]

\[ (\nu l)(\langle \bar{x}^l \cdot P | l[\langle \bar{M} \rangle ^\wedge \cdot Q]\) \cong (\nu l)(P\{\bar{x} := \bar{M}\} | l[Q]) \]

\[ m[(\bar{x}^l \cdot P | l[\langle \bar{M} \rangle ^\wedge \cdot Q]] \cong m[P\{\bar{x} := \bar{M}\} | l[Q]] \]

Mobility laws

\[ (\nu p)(m[\text{enter}\langle n, p \rangle \cdot P] | n[\text{enter}(x, p) \cdot Q]) \cong (\nu p)(n[Q\{x := m\} | m[P]]) \]

\[ l[m[\text{enter}\langle n, p \rangle \cdot P] | n[\text{enter}(x, p) \cdot Q]] \cong l[n[Q\{x := m\} | m[P]]] \]
An LTS for NBA

Concretions: \((\nu \tilde{p})(P)Q\) and \((\nu \tilde{p})(M)Q\)

\begin{align*}
\text{(Amb Co-enter)} & \quad P \xrightarrow{\text{enter}(n,k)} P' \\
& \quad m[P] \xrightarrow{\text{enter}(n,k)} (\nu)(P')0
\end{align*}

\begin{align*}
\text{(Co-enter HO)} & \quad P \xrightarrow{\text{enter}(n,k)} (\nu \tilde{p})(P_1)P_2 \quad \tilde{p} \cap \text{fn}(Q) = \emptyset \\
& \quad P \xrightarrow{\text{enter}(n,k)Q} (\nu \tilde{p})(m[n\{Q\} | P_1] | P_2)
\end{align*}

\begin{align*}
\text{(Exit)} & \quad P \xrightarrow{\text{exit}(n,k)} (\nu \tilde{p})(m[P_1])P_2 \\
& \quad n[P] \xrightarrow{\text{exit}(k)} (\nu \tilde{p})(m\{P_1\} \mid n[P_2])
\end{align*}

\begin{align*}
\text{(Exit HO)} & \quad P \xrightarrow{\text{exit}(n,k)} (\nu \tilde{p})(m[P_1])P_2 \quad x \in \text{fn}(R) \quad \tilde{p} \cap \text{fn}(Q|R) = \emptyset \\
& \quad P \xrightarrow{\text{exit}(n,k)QR} (\nu \tilde{p})(m[P_1] \mid n[P_2 \mid Q] \mid R\{x := m\})
\end{align*}
A Characterisation of Reduction Bisimulation

**Thm.** If \( P \xrightarrow{t} P' \) then \( P \rightarrow P' \). If \( P \rightarrow P' \) then \( P \xrightarrow{t} \equiv P' \).

**Bisimilarity.** A symmetric relation \( R \) is a bisimulation if

\[
P R Q \quad \text{and} \quad P \xrightarrow{\alpha} P' \quad \text{implies} \quad \exists Q \xrightarrow{\hat{\alpha}} Q' \quad \text{with} \quad P' R Q'.
\]

\( P \approx Q \) if \( P R Q \) for some bisimulation \( R \).

The closure under substitutions of \( \approx \) is denoted by \( \approx_c \).

**Thm.** If \( P \approx_c Q \) then \( P \equiv Q \) and viceversa.
A Type System for NBA

Types

Message Types

\[ W ::= N[E] \quad \text{ambient/password} \]
\[ | \quad C[E] \quad \text{capability} \]

Exchange Types

\[ E, F ::= \text{shh} \quad \text{no exchange} \]
\[ | \quad W_1 \ldots W_k \quad \text{tuples (}k \geq 0\text{)} \]

Process Types

\[ T ::= [E, F] \quad \text{composite exchange} \]

\( N[E] \) types both ambients and passwords; \text{shh} is the \text{silent type}; \( N[\text{shh}] \) is an ambient with no upward exchanges or a password that reveal the visitor’s name.
A Type System for NBA

Types

Message Types

\[ W ::= N[E] \quad \text{ambient/password} \]
\[ \quad | \quad C[E] \quad \text{capability} \]

Exchange Types

\[ E, F ::= \text{shh} \quad \text{no exchange} \]
\[ \quad | \quad W_1 \ldots W_k \quad \text{tuples} (k \geq 0) \]

Process Types

\[ T ::= [E, F] \quad \text{composite exchange} \]

\( N[E] \) types both ambients and passwords; \( \text{shh} \) is the silent type; \( N[\text{shh}] \) is an ambient with no upward exchanges or a password that reveal the visitor’s name.

Type Environments

\begin{align*}
\text{(Env Empty)} & \quad \text{(Env Name)} \\
\emptyset \vdash \diamond & \quad \Gamma \vdash \diamond \quad a \notin \text{Dom}(\Gamma) \\
\quad & \quad \Gamma, a : W \vdash \diamond
\end{align*}
Typing Rules

### Messages

**Projection**

\[ \Gamma, a : W, \Gamma' \vdash \diamond \]

\[ \Gamma, a : W, \Gamma' \vdash a : W \]

**Path**

\[ \Gamma \vdash M_1 : C[E_1] \quad \Gamma \vdash M_2 : C[E_2] \]

\[ \Gamma \vdash M_1 \cdot M_2 : C[E_1 \sqcup E_2] \]

**Enter**

\[ \Gamma \vdash M : N[E] \quad \Gamma \vdash N : N[F] \quad (F \leq G) \]

\[ \Gamma \vdash \text{enter}(M, N) : C[G] \]

**Exit**

\[ \Gamma \vdash M : N[E] \quad \Gamma \vdash N : N[F] \quad (F \leq G) \]

\[ \Gamma \vdash \text{exit}(M, N) : C[G] \]
Typing Rules

Messages

(projection)
\[ \Gamma, a : W, \Gamma' \vdash \Diamond \]
\[ \Gamma, a : W, \Gamma' \vdash a : W \]

(path)
\[ \Gamma \vdash M_1 : C[E_1] \quad \Gamma \vdash M_2 : C[E_2] \]
\[ \Gamma \vdash M_1 . M_2 : C[E_1 \uplus E_2] \]

(enter)
\[ \Gamma \vdash M : N[E] \quad \Gamma \vdash N : N[F] \quad (F \leq G) \]
\[ \Gamma \vdash \text{enter}(M, N) : C[G] \]

(exit)
\[ \Gamma \vdash M : N[E] \quad \Gamma \vdash N : N[F] \quad (F \leq G) \]
\[ \Gamma \vdash \text{exit}(M, N) : C[G] \]

Processes

(par)
\[ \Gamma \vdash P : [E, F] \quad \Gamma \vdash Q : [E, F] \]
\[ \Gamma \vdash P \mid Q : [E, F] \]

(repl)
\[ \Gamma \vdash P : [E, F] \]
\[ \Gamma \vdash \! P : [E, F] \]

(dead)
\[ \Gamma \vdash \Diamond \]
\[ \Gamma \vdash 0 : [E, F] \]

(new)
\[ \Gamma, n : N[G] \vdash P : [E, F] \]
\[ \Gamma \vdash (\nu n : N[G]) P : [E, F] \]
Typing Rules: II

Processes: mobility

(Amb)
\[ \Gamma \vdash M : N[E] \quad \Gamma \vdash P : [F, E] \]
\[ \Gamma \vdash M[P] : [G, H] \]

(Prefix)
\[ \Gamma \vdash M : C[F] \quad \Gamma \vdash P : [E, G] \quad (F \leq G) \]
\[ \Gamma \vdash M \cdot P : [E, G] \]

(Co-enter)
\[ \Gamma \vdash M : N[\tilde{W}] \quad \Gamma, x : N[\tilde{W}] \vdash P : [E, F] \]
\[ \Gamma \vdash \text{enter}(x, M).P : [E, F] \]

(Co-exit)
\[ \Gamma \vdash M : N[\tilde{W}] \quad \Gamma, x : N[\tilde{W}] \vdash P : [E, F] \]
\[ \Gamma \vdash \text{exit}(x, M).P : [E, F] \]

(Co-enter-silent)
\[ \Gamma \vdash M : N[\text{shh}] \quad \Gamma \vdash P : [E, F] \quad (x \not\in \text{fv}(P)) \]
\[ \Gamma \vdash \text{enter}(x, M).P : [E, F] \]

(Co-exit-silent)
\[ \Gamma \vdash M : N[\text{shh}] \quad \Gamma \vdash P : [E, F] \quad (x \not\in \text{fv}(P)) \]
\[ \Gamma \vdash \text{exit}(x, M).P : [E, F] \]
Typing Rules: II

Processes: I/O

**INPUT**

\[ \Gamma, \overline{x} : \overline{W} \vdash P : [\overline{W}, E] \quad \frac{\Gamma \vdash (\overline{x} : \overline{W}).P : [\overline{W}, E]}{} \]

**INPUT ^\wedge**

\[ \Gamma, \overline{x} : \overline{W} \vdash P : [E, \overline{W}] \quad \frac{\Gamma \vdash (\overline{x} : \overline{W})^\wedge P : [E, \overline{W}]}{} \]

**INPUT M**

\[ \Gamma \vdash M : N[\overline{W}] \quad \frac{\Gamma, \overline{x} : \overline{W} \vdash P : [G, H]}{} \quad \frac{\Gamma \vdash (\overline{x} : \overline{W})^M P : [G, H]}{} \]

**OUTPUT**

\[ \Gamma \vdash \overline{M} : \overline{W} \quad \frac{\Gamma \vdash P : [\overline{W}, E]}{} \quad \frac{\Gamma \vdash \langle \overline{M} \rangle . P : [\overline{W}, E]}{} \]

**OUTPUT ^\wedge**

\[ \Gamma \vdash \overline{M} : \overline{W} \quad \frac{\Gamma \vdash P : [E, \overline{W}]}{} \quad \frac{\Gamma \vdash \langle \overline{M} \rangle ^\wedge P : [E, \overline{W}]}{} \]

**OUTPUT N**

\[ \Gamma \vdash N : N[\overline{W}] \quad \frac{\Gamma \vdash \overline{M} : \overline{W}}{} \quad \frac{\Gamma \vdash P : [G, H]}{} \quad \frac{\Gamma \vdash \langle \overline{M} \rangle ^N P : [G, H]}{} \]

**Subject Reduction.** If \( \Gamma \vdash P : T \) and \( P \rightarrow Q \), then \( \Gamma \vdash Q : T \).
Encoding: BA in NBA

We can encode BA into NBA enriched with a focused form of nondeterminism.

\[
\begin{align*}
\{ P \}_n &= \text{cross} | \langle P \rangle_n \\
\langle m[P] \rangle_n &= m[\{ P \}_m] \\
\langle (x)^a P \rangle_n &= (x)^a \langle P \rangle_n \\
\langle (x) P \rangle_n &= (x) \langle P \rangle_n + (x)\hat{\langle P \rangle}_n + \overline{\text{exit}}(y, pw)(x)^y \langle P \rangle_n \\
\langle (x)^\dagger P \rangle_n &= (\nu p)p[\text{exit}(n, pr). (x)\hat{.}\text{enter}(n, p). (x)\hat{\langle P \rangle}_n] | \overline{\text{enter}}(y, p)(x)^y \langle P \rangle_n \\
\langle \langle M \rangle a P \rangle_n &= \langle M \rangle a \langle P \rangle_n \\
\langle \langle M \rangle P \rangle_n &= \langle M \rangle \langle P \rangle_n + \langle M \rangle\hat{\langle P \rangle}_n + \overline{\text{exit}}(y, pr)(M)^y \langle P \rangle_n \\
\langle \langle M \rangle^\dagger P \rangle_n &= (\nu p)p[\text{exit}(n, pw). (M)\hat{.}\text{enter}(n, p). (\cdot)^\hat{\langle P \rangle}_n] | \overline{\text{enter}}(y, p)(\cdot)^y \langle P \rangle_n
\end{align*}
\]

where \text{cross} = !\overline{\text{enter}}(x, mv) | !\overline{\text{exit}}(x, mv), \text{in } n = \text{enter}(n, mv), \text{and out } n = \text{exit}(n, mv).
Encoding: BA in NBA

We can encode BA into NBA enriched with a focused form of nondeterminism.

\[
\begin{align*}
\{P\}_n &= \text{cross} \mid \{P\}_n \\
\langle m[P] \rangle_n &= m[\{P\}_m] \\
\langle (x)^a P \rangle_n &= (x)^a \langle P \rangle_n \\
\langle (x)P \rangle_n &= (x) \langle P \rangle_n + (x)^{=} \langle P \rangle_n + \text{exit}(y, pw)(x)^y \langle P \rangle_n \\
\langle (x)^{\uparrow} P \rangle_n &= (v')p[\text{exit}(n, pr). (x)^{\hat{\cdot}}. \text{enter}(n, p). (x)^\hat{\cdot}] \mid \overline{\text{enter}}(y, p)(x)^y \langle P \rangle_n \\
\langle {\langle M \rangle}^a P \rangle_n &= {\langle M \rangle}^a \langle P \rangle_n \\
\langle {\langle M \rangle} P \rangle_n &= {\langle M \rangle} \langle P \rangle_n + {\langle M \rangle}^{\hat{\cdot}} \langle P \rangle_n + \text{exit}(y, pr) \langle M \rangle^y \langle P \rangle_n \\
\langle {\langle M \rangle}^{\uparrow} P \rangle_n &= (v')p[\text{exit}(n, pw). (M)^{\hat{\cdot}}. \text{enter}(n, p). (\cdot)^\hat{\cdot}] \mid \overline{\text{enter}}(y, p)(\cdot)^y \langle P \rangle_n
\end{align*}
\]

where \(\text{cross} = !\overline{\text{enter}}(x, mv) \mid !\text{exit}(x, mv),\) in \(n = \text{enter}(n, mv),\) and out \(n = \text{exit}(n, mv).\)

\textbf{Thm.} If \(P \xrightarrow{\tau} P'\) then \(\{P\} \xrightarrow{\tau} \geq \{P'\}.\)

If \(\{P\} \xrightarrow{\tau} Q,\) then \(\exists P \xrightarrow{\tau} P'\) with \(Q \geq \{P'\}.\)

If \(P\) and \(Q\) are single-threaded, then \(\{P\}_n \simeq \{Q\}_n\) implies \(P \simeq Q.\)
Conclusion and Future Work

- Type inference.
- Information flow analysis.
- Comparison with Seal calculus.
- Implementation.
- Logics.