Mobile Ambients – Syntax

\[ P, Q ::= (\nu n)P \]
\[ 0 \]
\[ P \mid Q \]
\[ !P \]
\[ M[P] \]
\[ M.P \]
\[ (x_1, \ldots, x_k).P \]
\[ \langle M_1, \ldots, M_k \rangle \]

\[ M ::= n \mid x \mid \text{in } M \mid \text{out } M \mid \text{open } M \]
\[ \mid M_1.M_2 \mid \varepsilon \]
Mobile Ambients – Semantics

Entering and exiting an ambient:

\[ n[\text{in} \; m \; . \; P \mid Q] \mid m[R] \rightarrow m[n[P \mid Q] \mid R] \]
\[ m[n[\text{out} \; m \; . \; P \mid Q] \mid R] \rightarrow n[P \mid Q] \mid m[R] \]

Opening an ambient:

\[ \text{open} \; n \; . \; P \mid n[Q] \rightarrow P \mid Q \]

Asynchronous communication:

\[ (x_1, \ldots, x_k).P \mid \langle M_1, \ldots, M_k \rangle \rightarrow P\{M_1 / x_1, \ldots, M_k / x_k \} \]
Implementation

- A monothread implementation is easy to write...
Implementation

- A monothread implementation is easy to write...
- ... but not very efficient...
Implementation

- A monothread implementation is easy to write...
- ... but not very efficient...
- ... and much more difficult to write in a distributed setting!
Implementation

- A monothread implementation is easy to write...
- ... but not very efficient...
- ... and much more difficult to write in a distributed setting!

⇒ A distributed abstract machine for safe ambients (Sangiorgi, Valente 2001) (for well-typed monothreaded safe ambients)
Equational theory
References

• A. D. Gordon and L. Cardelli, *Equational properties of mobile ambients*, FoSSaCS’99 (and MSCS)

• M. Merro and M. Hennessy, *Bisimulation congruences in safe ambients*, POPL’02

• M. Merro and F. Zappa Nardelli, *Bisimulation proof methods for mobile ambients*, ICALP’03
Another notion of barb (CG)

- Exhibition of a name:

\[ P \downarrow n \triangleq P \equiv (\nu \vec{m})(n[P'] | P'') \]

with \( n \notin \vec{m} \)
Another notion of barb (CG)

- Exhibition of a name:
  \[ P \downarrow n \triangleq P \equiv (\nu m)(n[P'] \mid P'') \]
  with \( n \notin m \)

- Convergence to a name:
  \[ P \downarrow n \triangleq P \rightarrow^* \downarrow n \]
Contextual equivalence (CG)

• Contextual equivalence:

\[ P \simeq Q \triangleq C[P] \downarrow n \iff C[Q] \downarrow n \]

for any name \( n \) and context \( C \) such that \( C[P] \) and \( C[Q] \) are closed.
Contextual equivalence (CG)

- Contextual equivalence:

\[ P \simeq Q \triangleq C[P] \downarrow n \iff C[Q] \downarrow n \]

for any name \( n \) and context \( C \) such that \( C[P] \) and \( C[Q] \) are closed.

- \( \simeq \) is a congruence and contains \( \equiv \)

- Proof technique: labelled transition system... much tougher than in \( \pi \)!
Examples

- Opening:

\[(\nu n)(n \parallel \mid \text{open } n.P) \simeq P \text{ if } n \notin \text{fn}(P)\]
Examples

- Opening:

\[(\nu n)(n[] \mid \text{open } n.P) \simeq P \text{ if } n \notin fn(P)\]

- Perfect firewall:

\[(\nu n)n[P] \simeq 0 \text{ if } n \notin fn(P)\]
Examples

- Opening:
  \[(\nu n)(n[] \mid \text{open } n.P) \simeq P \text{ if } n \notin fn(P)\]

- Perfect firewall:
  \[(\nu n)n[P] \simeq 0 \text{ if } n \notin fn(P)\]

- Firewall and agent:
  \[(\nu k \ k' \ k'')(\text{Agent} \mid \text{Firewall}) \simeq (\nu w)w[Q \mid P]\]
Reduction barbed congruence (MZN)

- A slightly modified ambient calculus (systems vs processes, replication of actions)...
Reduction barbed congruence (MZN)

- A slightly modified ambient calculus (systems vs processes, replication of actions)...
- **Reduction barbed congruence:**
  The largest symmetric relation over systems which is reduction closed (weakly), contextual and barb preserving (weakly).
Reduction barbed congruence (MZN)

• A slightly modified ambient calculus (systems vs processes, replication of actions)...

• **Reduction barbed congruence:**
  The largest symmetric relation over systems which is reduction closed (weakly), contextual and barb preserving (weakly).

• A labelled transition system...
Reduction barbed congruence (MZN)

- A slightly modified ambient calculus (systems vs processes, replication of actions)...
- **Reduction barbed congruence:**
The largest symmetric relation over systems which is reduction closed (weakly), contextual and barb preserving (weakly).
- A labelled transition system...
- A definition of bisimilarity...
Reduction barbed congruence (MZN)

- A slightly modified ambient calculus (systems vs processes, replication of actions)...
- **Reduction barbed congruence:**
  The largest symmetric relation over systems which is reduction closed (weakly), contextual and barb preserving (weakly).
- A labelled transition system...
- A definition of bisimilarity...
- Bisimilarity and barbed congruence coincide!
Expressivity
Motivations

- Theoretical interest: What makes the ambient calculus so expressive? What are the minimal constructs?
- To simplify future works by decreasing the number of cases to study.
- Find ideas and strategies for an implementation.
Some expressivity results

! vs rec

- !P can always be encoded as recX.(P | X)
- No converse encoding is known
- rec is probably “more” expressive than !
Some expressivity results

Busi, Zavattaro 2002

- Instead of looking at Turing machines, they consider the decidability of termination
- For a calculus with \(!\) or \(rec\)
- With or without movements (\(in\) and \(out\))
- With or without restriction (\(\nu\))
- Proofs are based on an encoding of RAMs and a reduction to well-structured transition systems
Some expressivity results
Some expressivity results

Boneva, Talbot 2003

• Consider the calculus (with replication) without open

• Reachability problem (given $P, Q$, does $P \rightarrow^* Q$ hold ?) is undecidable...

• ... but becomes decidable if we replace

\[ !P \equiv P \mid !P \]

with an oriented reduction rule:

\[ !P \rightarrow P \mid !P \]
Some expressivity results

Boneva, Talbot 2003

- Name-convergence problem (given $P, n$, does $P \downarrow n$ hold?) is undecidable for both versions
- Model-checking problem (against ambient logics) is undecidable for both versions
Expressiveness of Pure Ambients

- It has been shown (Cardelli and Gordon) that pure ambients are *Turing-powerful*. 
Expressiveness of Pure Ambients

- It has been shown (Cardelli and Gordon) that pure ambients are *Turing-powerful*.
- We show that the *Synchronous π-Calculus* can be encoded into pure ambients in a “satisfactory” way.
Expressiveness of Pure Ambients

- It has been shown (Cardelli and Gordon) that pure ambients are *Turing-powerful*.
- We show that the *Synchronous π-Calculus* can be encoded into pure ambients in a “satisfactory” way.
- Turing machines are not a satisfactory reference in a distributed and concurrent world.

⇒ better refer to π
Expressiveness of Pure Ambients

- It has been shown (Cardelli and Gordon) that pure ambients are *Turing-powerful*.
- We show that the *Synchronous π-Calculus* can be encoded into pure ambients in a “satisfactory” way.
- Turing machines are not a satisfactory reference in a distributed and concurrent world.

⇒ better refer to π
- We could encode π into pure ambients through Turing machines.
- However, the encoding would not be *compositional*. 
Pure Safe Ambients

They are the safe ambients, without communication primitives and rules.
Syntax:

\[
P ::= (\nu n) \ P \text{ restriction}
| \ 0 \text{ nil process}
| \ P \mid Q \text{ parallel composition}
| \ !P \text{ replication}
| \ n[P] \text{ ambient}
| \ Cap.P \text{ capability}
\]

\[
Cap ::= \text{ in } n \mid \text{ in } n \mid \text{ out } n \mid \text{ out } n \mid \text{ open } n \mid \text{ open } n
\]
Pure Safe Ambients

Entering and exiting an ambient:

\[
\begin{align*}
&n[\text{in } m \cdot P \mid Q] \mid m[\text{in } m \cdot R \mid S] \leftrightarrow m[n[P \mid Q] \mid R \mid S] \\
&m[n[\text{out } m \cdot P \mid Q] \mid \text{out } m \cdot R \mid S] \leftrightarrow n[P \mid Q] \mid m[R \mid S]
\end{align*}
\]

Opening an ambient:

\[
\text{open } n \cdot P \mid n[\text{open } n \cdot Q] 
\leftrightarrow P \mid Q
\]
Outline

• Introduction
  $\pi$-calculus $\leftrightarrow \pi_{esc}$-calculus $\leftrightarrow$ pure ambients
• Definition of the $\pi_{esc}$-calculus and operational correspondence with the $\pi$-calculus
• Encoding the $\pi_{esc}$-calculus in pure ambients and operational correspondence
• Final encoding and main result
• Conclusion and future work
Reminder: Synchronous $\pi$

Syntax:

$P ::= (\nu n) P$ restriction $M ::= n \in Name$

| 0 nil process | $x \in Var$ |
| $P | Q$ parallel composition |
| $!P$ replication |
| $\overline{M} \langle M' \rangle . P$ output |
| $M(x).P$ input |

Communication rule:

$\overline{n}\langle m \rangle .P | n(x).Q \longrightarrow P | Q\{m/x\}$
\( \pi_{esc} - \text{Calculus: Syntax} \)

Same syntax as the \( \pi \)-calculus, adding:

\[
P ::= \ldots \quad [n : S] \text{ explicit channel} \\
    \quad (\nu x : M) \ P \text{ explicit variable } (x \neq M) \\
\]

\[
S ::= \varepsilon \quad \text{empty channel} \\
    \quad S \mid S' \quad \text{parallel composition} \\
    \quad \langle M \rangle . P \quad \text{concretion} \\
    \quad (x) . P \quad \text{abstraction} \\
\]

Mobile Computing – p. 23
\( \pi_{esc} \): Operational Semantics

- Rules are of the form \( \sigma : P \rightleftharpoons P' \): a process \( P \) reduces to a process \( P' \) in the environment \( \sigma \) (\( = \) a substitution binding every free variable of \( P \)).
- Substituting a variable in a prefix by its value:

\[
\sigma : x\langle M' \rangle . P \rightleftharpoons \overline{M}\langle M' \rangle . P
\]

\[
\sigma : x(y) . P \rightleftharpoons M(y) . P
\]
\[ \pi_{esc}: \text{Operational Semantics} \]

- Output and input on a channel:

\[
\sigma: [n: S] | \overline{n} \langle M \rangle . P \quad \rightarrow \quad [n: S] | \langle M \rangle . P
\]

\[
\sigma: [n: S] | n(x). P \quad \rightarrow \quad [n: S] | (x). P
\]
\( \pi_{esc} \): Operational Semantics

- Effective communication in a channel, creation of a new variable and activation of the continuations:

\[
\begin{align*}
x \neq M \\
\sigma : [n : S | \langle M \rangle.P | (x).Q] & \rightarrow [n : S'] | P | (\nu x : M) Q
\end{align*}
\]
\[\pi_{esc}: \text{Operational Semantics}\]

- Effective communication in a channel, creation of a new variable and activation of the continuations:

\[
x \neq M
\]

\[
\sigma : [n : S \mid \langle M \rangle.P \mid (x).Q] \leftrightarrow [n : S] \mid P \mid (\nu x : M) Q
\]

- Integration of a variable in the environment:

\[
x \notin \text{dom}(\sigma) \quad \{\frac{M/x}{\text{dom}(\sigma)}\} \uplus \sigma : P \leftrightarrow P'
\]

\[
\sigma : (\nu x : M) P \leftrightarrow (\nu x : M) P'
\]

- Reduction under \((\nu n)\), in parallel or by structural congruence \(\equiv\)...
Valid Processes and Channel Closure

- Channels can be unreachable: \( \overline{n}(m).[p : S] \) or too numerous:
  \( [n : S] \mid [n : S'] \mid \overline{n}(m).P \mid n(x).Q \)

\( \Rightarrow \) A simple type system to avoid those *invalid* processes. Validity is preserved by reduction.
Valid Processes and Channel Closure

- Channels can be unreachable: \( \overline{n}(m).[p : S] \)
or too numerous:
\[
[n : S] \mid [n : S'] \mid \overline{n}(m).P \mid n(x).Q
\]
⇒ A simple type system to avoid those invalid processes. Validity is preserved by reduction.

- Channels can be missing:
\[
(\nu n) (\overline{n}(m).P \mid n(x).Q)
\]
⇒ A channel closure (w.r.t. a substitution \( \sigma \)) \( cl_{\sigma}(P) \) to add missing channels. \( P \) is channel-closed if all channels are present.
From $\pi_{esc}$ to $\pi$

Translating a $\pi_{esc}$-process in a intuitively “equivalent” $\pi$-process:

$$[[n : S]] \triangleq [S]_n$$

$$[(\nu x : M) \ P] \triangleq [P] \{M/x\}$$

$$[\varepsilon]_n \triangleq 0$$

$$[S | S']_n \triangleq [S]_n | [S']_n$$

$$[(M).P]_n \triangleq \overline{n}(M).[P]$$

$$[(x).P]_n \triangleq n(x).[P]$$

([.] is an homomorphism for all other constructs)
Operational Correspondence $\pi_{esc} \rightarrow \pi$

**Proposition 1** If $\emptyset : P \leftrightarrow Q$, then $[P] \mathcal{R} [Q]$, where $\mathcal{R}$ is either $\equiv$ or $\rightarrow$.

<table>
<thead>
<tr>
<th>$\pi$-calculus</th>
<th>$\pi_{esc}$-calculus</th>
</tr>
</thead>
<tbody>
<tr>
<td>$[P]$</td>
<td>$P$</td>
</tr>
<tr>
<td>$\mathcal{R}$</td>
<td>$\downarrow$</td>
</tr>
<tr>
<td>$[Q]$</td>
<td>$Q$</td>
</tr>
</tbody>
</table>
Operational Correspondence $\pi \rightarrow \pi_{esc}$

**Proposition 2** If a process $P$ is channel-closed w.r.t. $\emptyset$, valid and without free variables, and if $[P] \rightarrow Q$, then there is a process $P'$ such that $\emptyset : P \rightarrow^+ P'$ and $[P'] \equiv Q$.

\[ \begin{array}{c c}
\pi\text{-calculus} & \Rightarrow & \pi_{esc}\text{-calculus} \\
[P] & \downarrow & P' \\
\vdots & \downarrow & \vdots \\
Q \equiv [P'] & \downarrow & P'
\end{array} \]
Encoding $\pi_{esc}$ into Pure Ambients

- Actors “communicate” by a request/server mechanism:
  - A server is a replicated process which tries to inject its code into requests and take their control.
  - A request is an ambient allowing the code injection and execution.
- A channel is simulated by an ambient $n$ receiving and processing read and write requests.
- A variable is simulated by an ambient $x$ receiving and processing read and write requests by forwarding them to $M$. 
Operational Correspondence

Proposition 3  If $\sigma : P \hookrightarrow Q$, then
$$\{\sigma, P\} \xrightarrow{pr} \xrightarrow{aux^*} \{\sigma, Q\}.$$  

Proposition 4  If $\{\sigma, P\} \xrightarrow{pr} Q$, then there is a process $P'$ such that $\sigma : P \hookrightarrow P'$ and $Q \xrightarrow{aux^*} \{\sigma, P'\}$. Moreover, if $\sigma : P \hookrightarrow P''$ and $Q \xrightarrow{aux^*} \{\sigma, P''\}$, then $P' \equiv P''$ (in other words $P'$ is unique modulo $\equiv$).
Encoding $\pi$ into Pure Ambients

- From the $\pi$-calculus to the $\pi_{esc}$-calculus: we only need to add channels, $(\nu n) \ P$ becomes $(\nu n) \ ([n : \varepsilon] \mid P)$.
- From the $\pi_{esc}$-calculus to pure ambients: $P$ becomes $\{\emptyset, P\}$.
- The final encoding $\langle\langle P\rangle\rangle$ is the composition of the two previous encodings.
- It can be written directly, and not via the $\pi_{esc}$-calculus...
Main Result

Definition  Let $P$ be a $\pi$-process with no free variables and $R$ a pure ambient process. We will say that $P$ and $R$ are equivalent (written $P \approx R$) if there is a $\pi_{esc}$-process $Q$ such that $Q$ is valid, channel-closed w.r.t. $\emptyset$, with no free variables, $P \equiv \llbracket Q \rrbracket$ and $\{ \emptyset, Q \} \equiv R$.

It is routine to check that $P \approx \llangle P \rrangle$ for every $\pi$-process $P$ with no free variables.

Theorem  Suppose $P \approx R$.

- If $P \xrightarrow{} P'$, then there is a process $R'$ such that $R \xrightarrow{pr} R'$ and $P' \approx R'$.
- If $R \xleftarrow{pr} R'$, then there is a process $R''$ such that $R' \xrightarrow{aux} R''$, and either $P \approx R''$ or $P \xrightarrow{} P' \approx R''$. 
Open Problems

- Proving a conjecture (with the help of an automatic demonstration tool) and state a stronger result for the operational correspondence
- Encoding the polyadic π-calculus (should be easy)
- Encoding the π-calculus in classical ambients instead of safe ambients (difficult ???)
- Main question: encoding ambients with communications into ambients without communications