Static Analysis Symposium 2001

Abstract Interpretation-Based Static Analysis of Mobile Ambients

Jérôme Feret
École normale supérieure
http://www.di.ens.fr/∼feret

July 18, 2001

Overview

1. mobile systems:
   (a) mobile systems,
   (b) mobile ambients;
2. non-standard semantics:
   (a) markers,
   (b) non-standard configuration;
3. abstract semantics:
   (a) abstract interpretation,
   (b) generic abstract semantics,
   (c) three instantiations.

MOBILE SYSTEMS

Mobile system

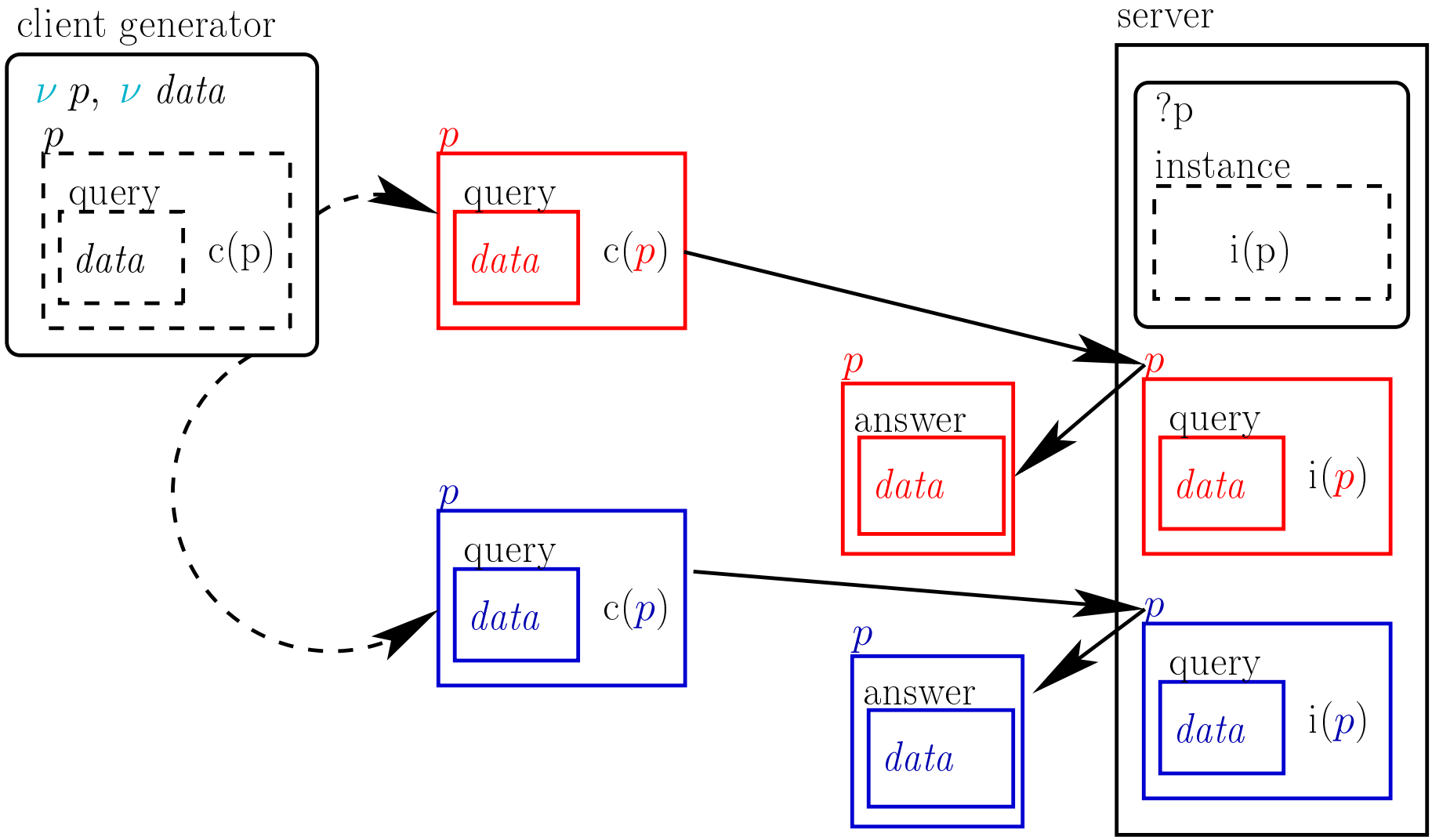
Mobile computation involves interacting agents distributed throughout hierarchically organized domains.

The system topology dynamically changes via:

- agents, names, domains creation;
- name communication (*implicit mobility*);
- domain migration (*explicit mobility*).

Topology of interaction may be unbounded !

An *ftp*-server



Mobile Ambients

Ambients are named boxes containing other ambients (and/or) some agents.

Agents:

- provide capabilities to their surrounding ambients for local migration and other ambient dissolution;
- dynamically create new ambients, names and agents;
- communicate names to each others.

Syntax

Let $\mathcal{N}$ be an infinite countable set of ambient names and $\mathcal{L}$ an infinite countable set of labels.

$$
\begin{array}{rll}
n & \in \mathcal{N} & \text{(ambient name)} \\
l & \in \mathcal{L} & \text{(label)}
\end{array}
$$

$$
\begin{array}{rcll}
P, Q & ::= & (\nu\ n)P & \text{(restriction)} \\
 & | & \mathbf{0} & \text{(inactivity)} \\
 & | & P \mid Q & \text{(composition)} \\
 & | & n^{l}[P] & \text{(ambient)} \\
 & | & M & \text{(capability action)} \\
 & | & io & \text{(input/output action)}
\end{array}
$$

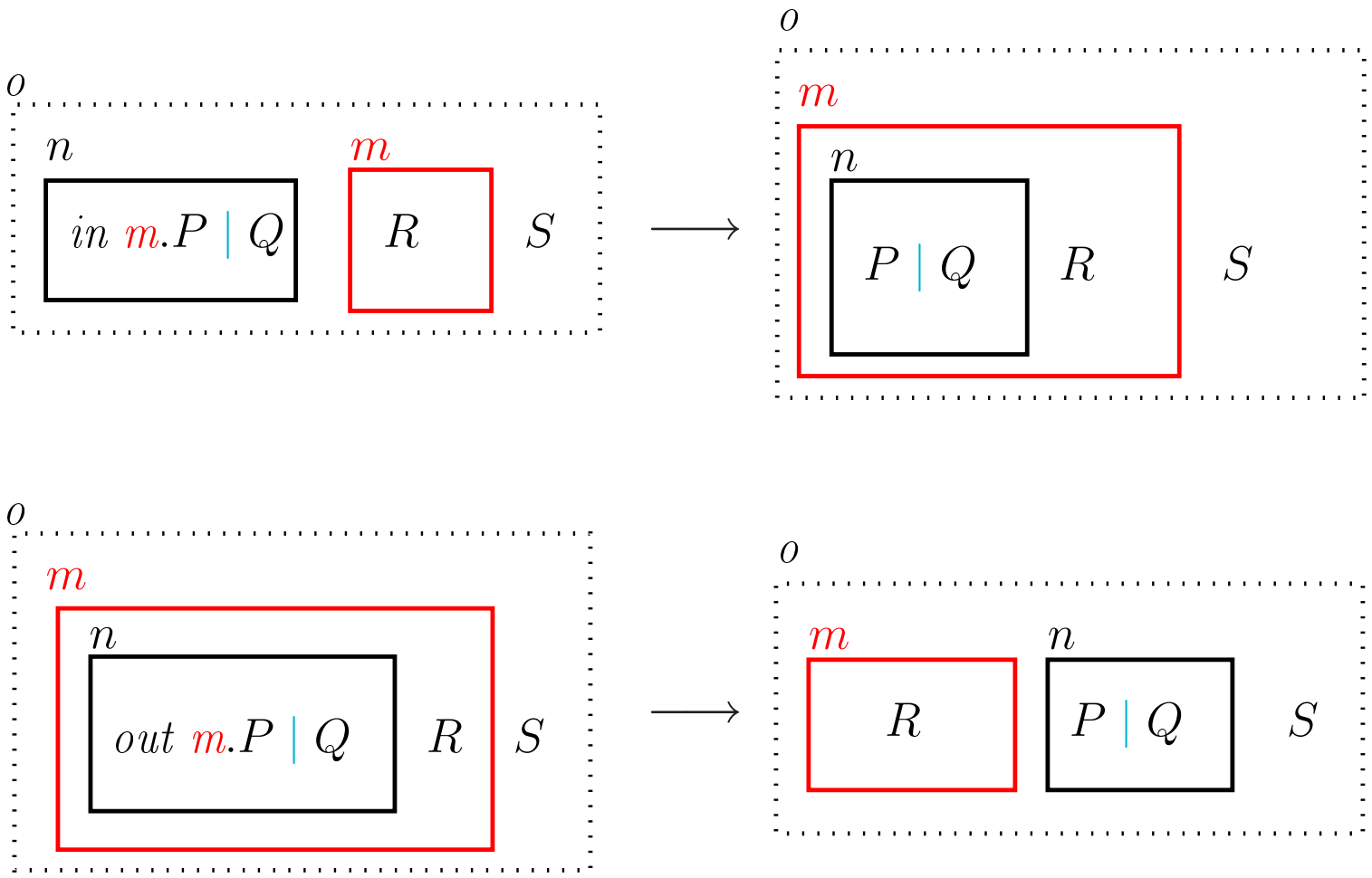
Capability and actions

$$
\begin{array}{rcll}
M & ::= & in^{l}\ n.P & \text{(can enter an ambient named } n\text{)} \\
 & | & out^{l}\ n.P & \text{(can exit an ambient named } n\text{)} \\
 & | & open^{l}\ n.P & \text{(can open an ambient named } n\text{)} \\
 & | & !open^{l}\ n.P & \text{(can open several ambients named } n\text{)}
\end{array}
$$

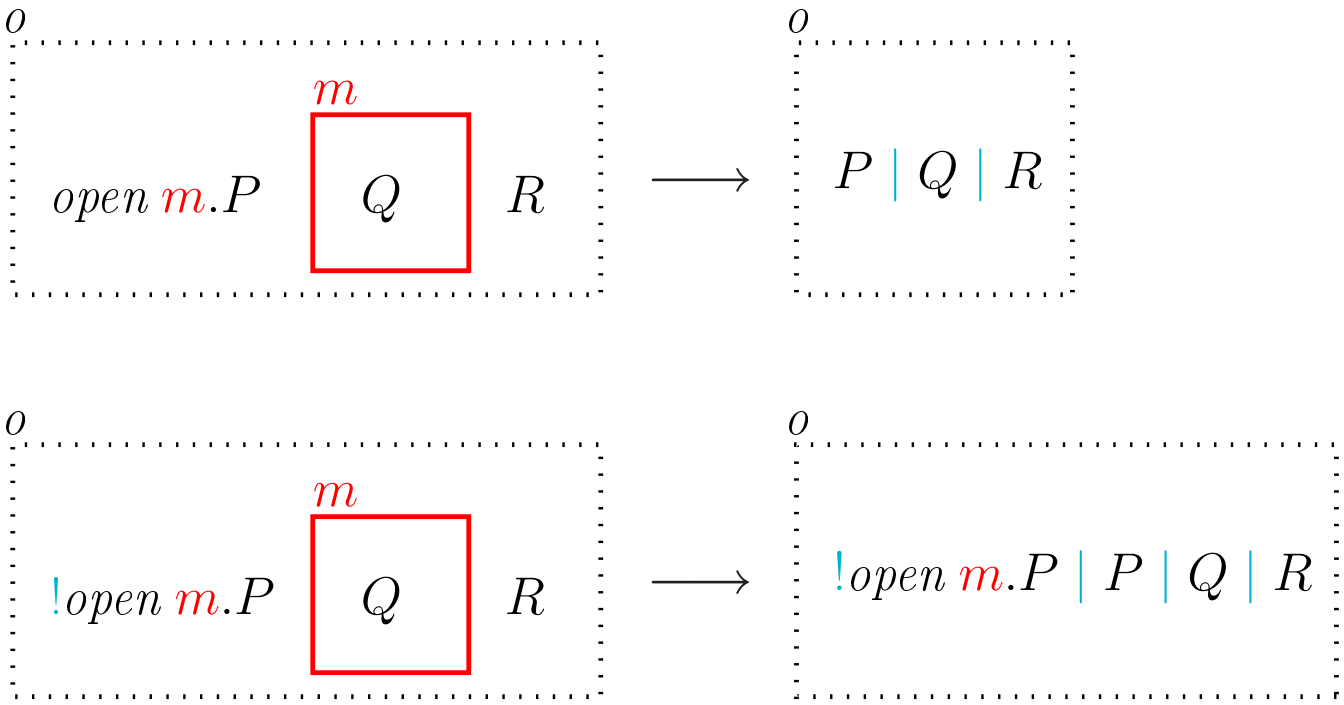
$$
\begin{array}{rcll}
io & ::= & (n)^{l}.P & \text{(input action)} \\
 & | & !(n)^{l}.P & \text{(input action with replication)} \\
 & | & \langle n \rangle^{l} & \text{(async output action)}
\end{array}
$$

The only name binders are $(\nu\ \_)$, $(\_)$ and $!(\_)$.

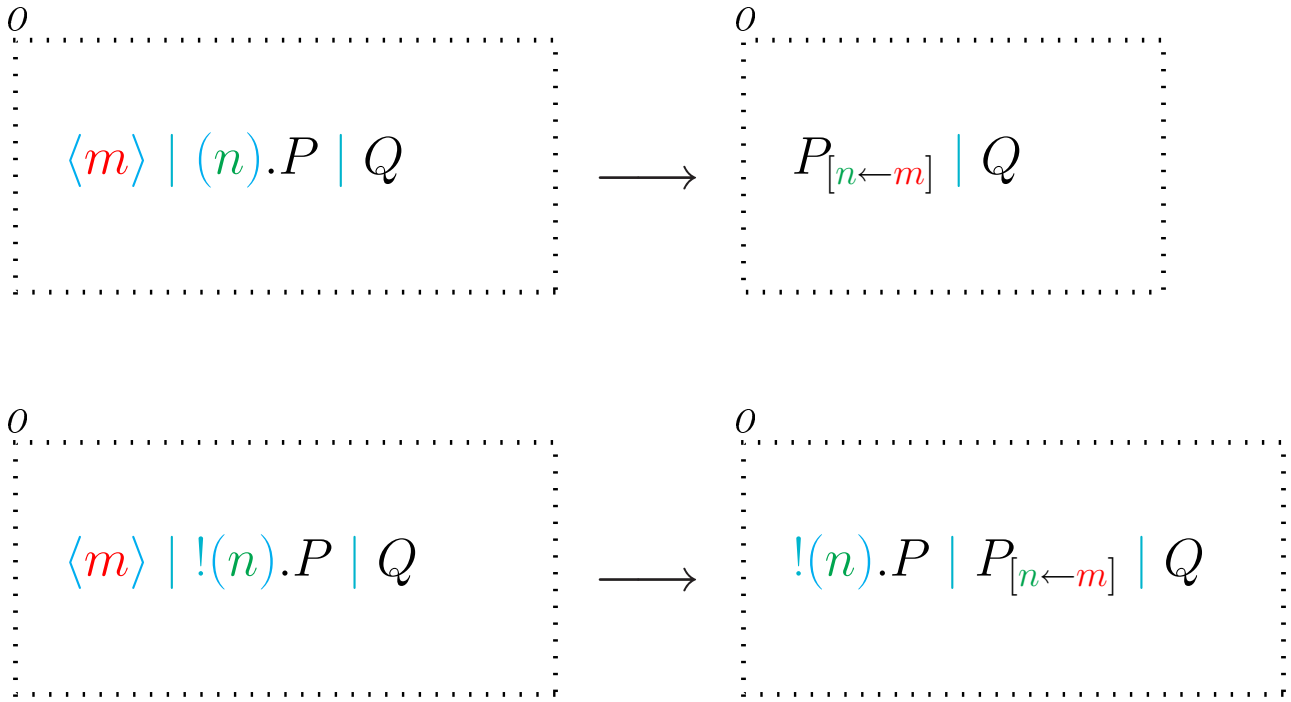
Ambient Migration



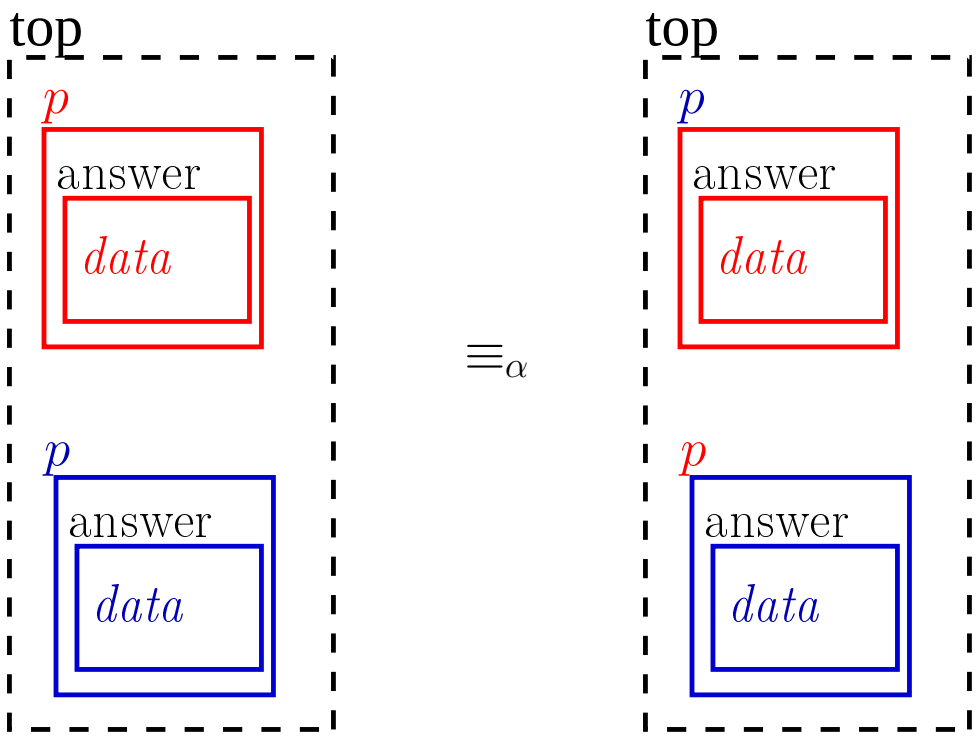
Ambient Dissolution



Communication



α-conversion



NON-STANDARD SEMANTICS

Explicit Link

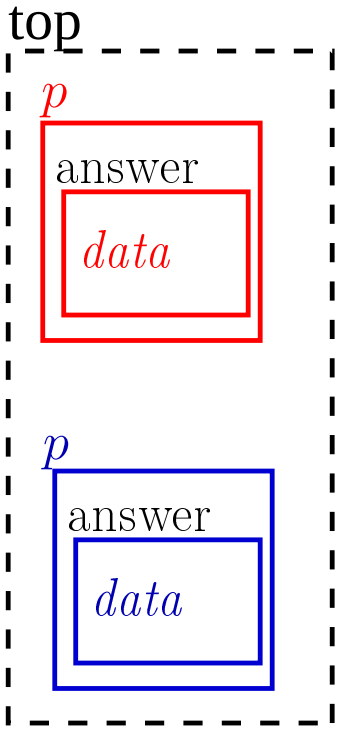
We propose to explicitly restore the link between recursive instances of agents and both the ambients and the names they have created:

1. we first associate an unambiguous marker to each instance of agent,
2. we then stamp each ambient and each name with the marker of the instance which has declared it.

Non-Standard Configuration

We flatly represent system configurations:

$$\left\{\begin{array}{l}(p^{12}[\bullet], id_0, (\mathrm{top}, \varepsilon), [p \mapsto (p, id_0)]) \\ (p^{12}[\bullet], id_1, (\mathrm{top}, \varepsilon), [p \mapsto (p, id_1)]) \\ (\mathrm{answer}^{8}[\bullet], id'_0, (12, id_0), \emptyset) \\ (\mathrm{answer}^{8}[\bullet], id'_1, (12, id_1), \emptyset) \\ \hline (\langle rep \rangle^{9}, id'_0, (8, id'_0), [rep \mapsto (data, id_0)]) \\ (\langle rep \rangle^{9}, id'_1, (8, id'_1), [rep \mapsto (data, id_1)])\end{array}\right.$$



Marker Properties

1. Marker allocation must be consistent:

   Two instances of the same agent must not be associated the same marker during a computation sequence.

2. Marker allocation should be robust:

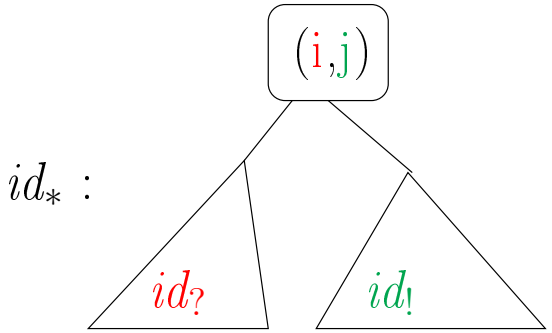
   Marker allocation should not depend on which order commuting computation steps are performed, otherwise we cannot capture any properties during the analysis.

History Markers

Markers describe the history of the replications which have led to the creation of the agent.
They are binary trees:

- leaves are not labeled;
- nodes are labeled with a pair $(i, j) \in \mathit{Label}^2$.

They are recursively calculated when fetching resources as follows:



Marker simplification

We can simplify marker shape without losing our semantics consistency:

- $\phi_1 : \left\{ \begin{array}{ccc} Id & \longrightarrow & (Label^2)^* \\ Node((i,j),l,r) & \mapsto & (i.j).\phi_1(r) \\ \varepsilon & \mapsto & \varepsilon; \end{array} \right.$
- $\phi_2 : \left\{ \begin{array}{ccc} Id & \longrightarrow & Label^* \\ Node((i,j),l,r) & \mapsto & j.\phi_2(r) \\ \varepsilon & \mapsto & \varepsilon. \end{array} \right.$

Therefore, partitioning will be less precise when abstracting.

ABSTRACT SEMANTICS

Collecting Semantics

Since we only focus on the invariants satisfy at any stage our mobile system may take during a finite computation sequence, we can restrict our study to its collecting semantics:

$$\mathcal{S}(C_0) = \{C' \mid C_0 \rightarrow^* C'\}$$

which is also the least fix-point of a $\cup$-complete endomorphism:

$$\mathcal{S}(C_0) = \mathit{lfp}_{\emptyset}\mathbb{F}$$
$$\text{where } \mathbb{F} \ : \ X \mapsto \{C_0\} \cup \{C' \mid \exists C \in X,\ C \rightarrow C'\}$$

Abstract Properties

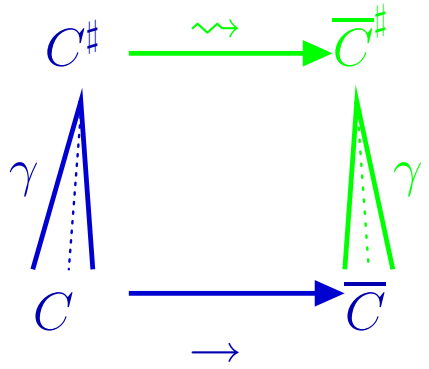
We introduce an abstract domain of properties:

- properties of interest;
- more complex properties required in calculating them.

This domain is often a binary lattice: $(\mathcal{D}^\sharp, \sqsubseteq, \sqcup, \bot, \sqcap, \top)$ and is related to the concrete domain $\wp(\mathcal{C})$ by a monotonic concretization function γ.
$\forall A \in \mathcal{D}^\sharp$, $\gamma(A)$ is the set of the elements which satisfies the property A.

Abstract transition system

We introduce an abstract transition system $(C_0^\sharp, \rightsquigarrow)$, such that $C_0 \subseteq \gamma(C_0^\sharp)$ and that the following diagram is satisfied:



$$\text{Then } \mathcal{S} \subseteq \bigcup_{n \in \mathbb{N}} \gamma(\mathbb{F}^{\sharp n}(C_0^\sharp))$$

$$\text{where } \mathbb{F}^\sharp(C^\sharp) = C_0^\sharp \sqcup C^\sharp \sqcup \left(\bigsqcup_{\textit{finite}} \{\overline{C^\sharp} \mid C^\sharp \rightsquigarrow \overline{C^\sharp}\} \right).$$

Abstract Semantics

We abstract:

- for each name restriction $(\nu\ x)$ and each variable y of a syntactic agent P, the set of the marker pairs $(id_P.id_x)$ such that a name created by the instance of (νx) the marker of which is id_x may be communicated to the variable y of the instance of the agent P the marker of which is id_P;
- for each syntactic agent P and each ambient activator $\_^i[\_]$, the set of the marker pairs $(id_P.id_i)$ such that the instance of P the marker of which is (id_P) may be located in the ambient created by the instance of the ambient activator $\_^i[\_]$ the marker of which is id_i.

0-CFA Analysis

We choose the lattice $(\{\bot, \top\}, \sqsubseteq)$ with $\bot \subseteq \top$, related to $\wp(Id^2)$ by the concretization function γ, defined by $\gamma(\bot) = \emptyset$ and $\gamma(\top) = Id^2$.

This analysis computes sound approximative answers to the questions:

- May a name created by an instance of the name restriction $(\nu\ x)$ be communicated to the variable y of an instance of the thread P ?
- May an instance of the thread P be located in an ambient created by an instance of the ambient activator $\_^i[\_]$?

Domain complexity is $O(1)$ and maximum iteration number is $O(n^3)$.

Confinement Analysis

We choose the lattice $(\{\bot, =, \top\}, \sqsubseteq)$ with $\bot \subseteq = \subseteq \top$, related to $\wp(Id^2)$ by the concretization function γ, defined by :

$$\gamma(\bot) = \emptyset;\ \gamma(=) = \{(id, id) \mid id \in Id\};\ \gamma(\top) = Id^2.$$

This analysis also computes sound approximative answers to the questions:

- Is a name confined in the scope of the recursive instance which has declared it ?
- Is a thread always located in an ambient declared by the same recursive instance than the one which has spawn this thread ?

Domain complexity is $O(1)$ and maximum iteration number is $O(n^3)$.

Non-Uniform Analysis

Non-uniform analysis allows us to express a wider class of properties:

- may a name be only communicated to the next instance of the resource which has created it;
- may a name be only communicated to the previous instance of the resource which has created it.

This class is complete enough to capture interesting properties.
We use the product of two domains:

- a regular approximation;
- a numerical approximation.

Regular Approximation

We approximate the shape of each word in the following abstract domain:

$$\wp(\Sigma) \times \wp(\Sigma) \times \wp(\Sigma \times \Sigma) \times \{\mathit{true};\mathit{false}\}.$$

$\gamma(I, F, T, b)$ is defined by $\gamma_1(I) \cap \gamma_2(F) \cap \gamma_3(T) \cap \gamma_4(b)$ where:

- $\gamma_1(I) = \{L \mid \forall u \in L,\ |u| > 0 \Rightarrow u_1 \in I\}$,
- $\gamma_2(F) = \{L \mid \forall u \in L,\ |u| > 0 \Rightarrow u_{|u|} \in F\}$,
- $\gamma_3(T) = \{L \mid \forall u, v \in \Sigma^*, \lambda, \mu \in \Sigma,\ u.\lambda.\mu.v \in L \Rightarrow (\lambda, \mu) \in T\}$,
- $\gamma_4(b) = \{L \mid \varepsilon \in L \Rightarrow b\}$.

Domain complexity is $O(n.|\Sigma|)$ and maximum iteration number is $O(n^4.|\Sigma|)$.

Affine Approximation

We capture the difference between the occurrence number of letters in the first and the second component of marker pairs:

$$(\mathcal{F}(\Sigma, \mathbb{N} \cup \{\top\})) \cup \{\bot\}$$

γ is defined as follows:

$\gamma(\bot) = \emptyset$

$\gamma(f) = \{L \in \Sigma^2 \mid \forall \lambda \in \Sigma,\ f(\lambda) \in \mathbb{N} \Rightarrow \forall (u,v) \in L,\ |u|_\lambda - |v|_\lambda = f(n)\}$.

Domain complexity is $O(|\Sigma|)$ and maximum iteration number is $O(n^3.|\Sigma|)$.

Example

We detect that:

$$\begin{cases} (p^{12}[\bullet], (11,20)^m.(11,21), \_, [p \mapsto (p, (11,20)^m.(11,21))]) \\ (\text{answer}^8[\bullet], (3,19).(11,20)^n.(11,21), (12, (11,20)^n.(11,21), \_) \\ (\langle rep \rangle^9, \_, (8, (3,19).(11,20)^p.(11,21), [rep \mapsto (data, (11,20)^p.(11,21))])) \end{cases}$$

We deduce that each packet exiting the server has the following structure:

$(p.(11,20)^n.(11,21))$

answer

$(data, (11,20)^n.(11,21))$ $(3,19).(11,20)^n.(11,21)$

$(11,20)^n.(11,21)$

Conclusion

- We have infered a sound but not complete description of the interactions between the agents of a mobile ambients.
- Our analysis is highly generic. We have proposed three instantiations in accordance to the trade off between accuracy and complexity.
- Our analysis has succeeded in proving the integrity of an *ftp*-server, described in the ambient calculus.

Future Work

- Design a shape analysis,

  $\Rightarrow$ to refine our analysis by detecting mutual exclusion;

- Propose a worst-case semantics for analyzing a small part of a system, without having much knowledge about the rest of it.

  $\Rightarrow$ to design a modular analysis;

- Infer behavioral properties expressed in modal logic, which handle with dynamic declaration of both ambient names and ambients.

  $\Rightarrow$ to prove user-friendly properties.