Static Analysis Symposium 2000

Confidentiality analysis for mobile systems

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

July 1, 2000

Mobile systems

Mobile system

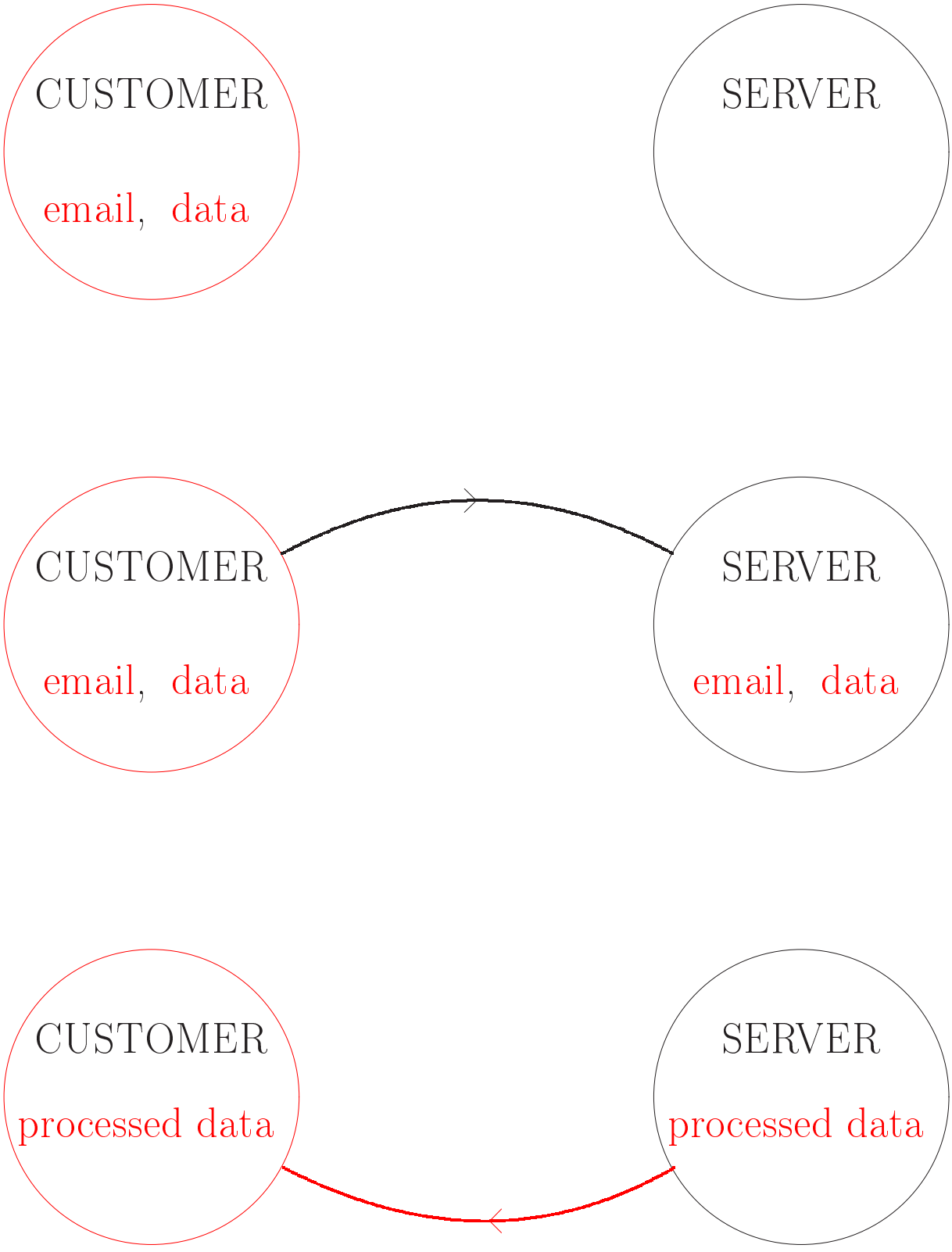
A pool of processes which interact via communications:

Communications allow to

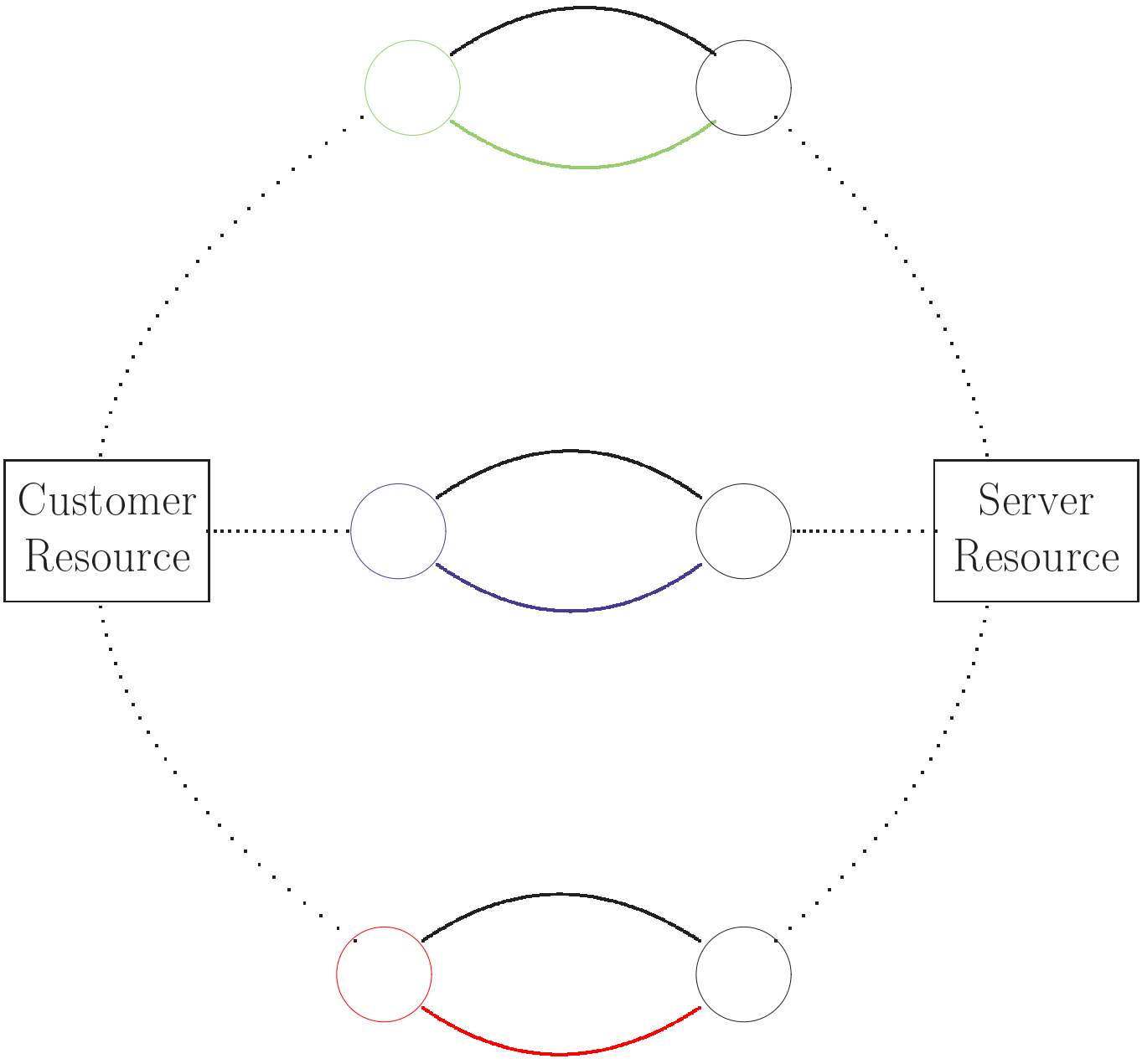
- synchronize process computation;
- change structure of processes;
- create new communication links;
- create new processes.

Topology of interaction may be unbounded !

A connexion:


CUSTOMER
email, data
SERVER
CUSTOMER
email, data
SERVER
email, data
CUSTOMER
processed data
SERVER
processed data

A network:



Confidentiality

A sound description of the communication topology allows to prove that private information cannot be passed to forbidden processes.

Objectives

- Non-uniform analysis:
  - distinguishing recursive instance of processes;
- No particular assumptions on mobile systems:
  - considering untyped languages;
  - allowing embedded resources.
- Context-free analysis:
  - analyzed systems belong to bigger unknown systems.

π-calculus : syntax

Let *Channel* be an infinite set of channel names, and *Label* an infinite set of labels,

$$
\begin{array}{rcll}
P & ::= & \text{action}.P & \text{(Action)} \\
 & | & (P \mid P) & \text{(Parallel composition)} \\
 & | & \emptyset & \text{(End of a process)}
\end{array}
$$

$$
\begin{array}{rcll}
\text{action} & ::= & c!^{i}[x_1, ..., x_n] & \text{(Message)} \\
 & | & c?^{i}[x_1, ..., x_n] & \text{(Input guard)} \\
 & | & *c?^{i}[x_1, ..., x_n] & \text{(Replication guard)} \\
 & | & (\nu\ x) & \text{(Channel creation)}
\end{array}
$$

where $n \geqslant 0$,
$c,\ x_1,\ ...,\ x_n,\ x,\ \in$ *Channel* and $i \in$ *Label*.

ν and $?$ are the only name binders. We denote by $\mathcal{FN}(P)$ the set of free names in P, and by $\mathcal{BN}(P)$ the set of bound names in P.

Transition semantics

A reduction relation and a congruence relation give the semantics of the π-calculus:

- the reduction relation specifies results of process computation:

$$c?^i[\overline{y}]Q \mid c!^j[\overline{x}]P \xrightarrow{i,j} Q[\overline{y} \leftarrow \overline{x}] \mid P$$
$$*c?^i[\overline{y}]Q \mid c!^j[\overline{x}]P \xrightarrow{i,j} Q[\overline{y} \leftarrow \overline{x}] \mid *c?^i[\overline{y}]Q \mid P$$

- the congruence relation reveals redexs:
  - names renaming (α-conversion),
  - structural modifications (Commutativity, associativity, and so on).

Example: syntax

$$\mathcal{S} := (\nu\ \text{port})(\nu\ \text{gen})\ (\textbf{Server} \mid \textbf{Customer} \mid \text{gen}!^{0}[])$$

where

$$\textbf{Server} := *\text{port}?^{1}[\mathit{info}, \mathit{add}](\mathit{add}!^{2}[\mathit{info}])$$

$$\textbf{Customer} := *\text{gen}?^{3}[]\ ((\nu\ \mathit{data})\ (\nu\ \mathit{email})\ (\text{port}!^{4}[\mathit{data}, \mathit{email}] \mid \text{gen}!^{5}[]))$$

Example: computation

$$\begin{aligned}
\textbf{Server} &:= *\text{port}?^{1}[info, add](add!^{2}[info]) \\
\textbf{Customer} &:= *\text{gen}?^{3}[]\ ((\nu\ data)(\nu\ email) \\
&\qquad (\text{port}!^{4}[data, email]\ \mid\ \text{gen}!^{5}[]))
\end{aligned}$$

$$\begin{aligned}
&(\nu\ \text{port})(\nu\ \text{gen}) \\
&\ (\textbf{Server}\ \mid\ \textbf{Customer}\ \mid\ \text{gen}!^{0}[])
\end{aligned}$$

$$\begin{aligned}
\xrightarrow{3,0}\ &(\nu\ \text{port})(\nu\ \text{gen})(\nu\ data_1)(\nu\ email_1) \\
&\ (\textbf{Server}\ \mid \textbf{Customer} \mid \text{gen}!^{5}[] \\
&\ \mid\ \text{port}!^{4}[data_1, email_1])
\end{aligned}$$

$$\begin{aligned}
\xrightarrow{1,4}\ &(\nu\ \text{port})(\nu\ \text{gen})(\nu\ data_1)(\nu\ email_1) \\
&\ (\textbf{Server}\ \mid\ \textbf{Customer}\ \mid\ \text{gen}!^{5}[] \\
&\ \mid\ email_1!^{2}[data_1])
\end{aligned}$$

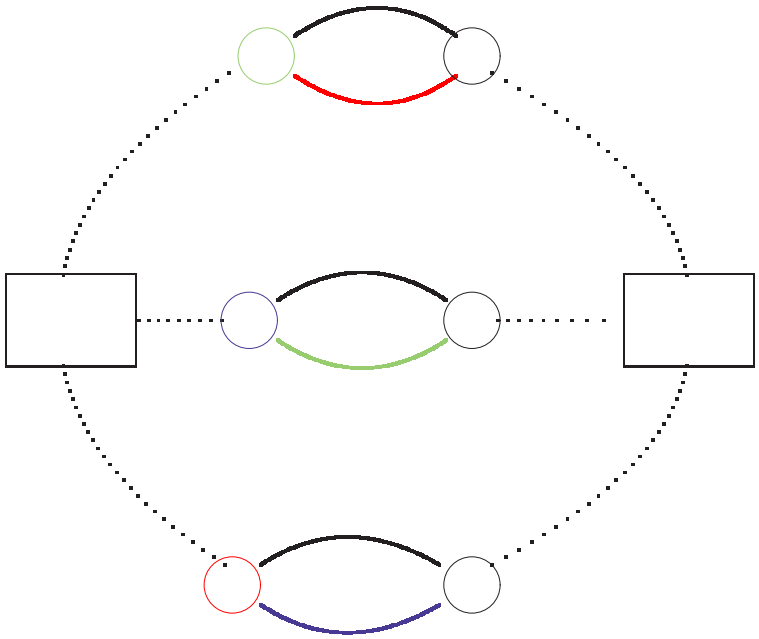
$$\begin{aligned}
\xrightarrow{3,0}\ &(\nu\ \text{port})(\nu\ \text{gen})(\nu\ data_1)(\nu\ email_1)(\nu\ data_2)(\nu\ email_2) \\
&\ (\textbf{Server}\ \mid\ \textbf{Customer}\ \mid\ \text{gen}!^{5}[] \\
&\ \mid\ email_1!^{2}[data_1]\ \mid\ \text{port}!^{4}[data_2, email_2])
\end{aligned}$$

$$\begin{aligned}
\xrightarrow{1,4}\ &(\nu\ \text{port})(\nu\ \text{gen})(\nu\ data_1)(\nu\ email_1)(\nu\ data_2)(\nu\ email_2) \\
&\ (\textbf{Server}\ \mid\ \textbf{Customer}\ \mid\ \text{gen}!^{5}[] \\
&\ \mid\ email_1!^{2}[data_1]\ \mid\ email_2!^{2}[data_2])
\end{aligned}$$

α-conversion

α-conversion destroys the link between channel names and processes which have declared them:

$$(\nu\ \text{port})(\nu\ \text{gen})(\nu\ data_1)(\nu\ email_1)(\nu\ data_2)(\nu\ email_2)$$
$$(\textbf{Server} \mid \textbf{Customer} \mid \text{gen}!^5[]$$
$$\mid\ email_1!^4[data_1] \mid email_2!^4[data_2])$$
$$\sim_\alpha$$
$$(\nu\ \text{port})(\nu\ \text{gen})(\nu\ data_2)(\nu\ email_1)(\nu\ data_1)(\nu\ email_2)$$
$$(\textbf{Server} \mid \textbf{Customer} \mid \text{gen}!^5[]$$
$$\mid\ email_1!^4[data_2] \mid email_2!^4[data_1])$$



Non-standard semantics

Non-standard semantics

A refined semantics in where

- recursive instances of processes are identified with unambiguous markers;
- channel names are enriched with the marker of the process which has declared them.

Example: non-standard configuration

(**Server** | **Customer** | $\text{gen}!^{5}[]$ | $email_1!^{2}[data_1]$ | $email_2!^{2}[data_2]$)

$$\left\{ \begin{array}{l} \left(1, \varepsilon, \left\{ \text{port} \mapsto (\text{port}, \varepsilon) \right\}\right) \\ \left(3, \varepsilon, \left\{ \begin{array}{ll} \text{gen} & \mapsto (\text{gen}, \varepsilon) \\ \text{port} & \mapsto (\text{port}, \varepsilon) \end{array} \right\}\right) \\ \left(2, id'_1, \left\{ \begin{array}{ll} add & \mapsto (email, id_1) \\ info & \mapsto (data, id_1) \end{array} \right\}\right) \\ \left(2, id'_2, \left\{ \begin{array}{ll} add & \mapsto (email, id_2) \\ info & \mapsto (data, id_2) \end{array} \right\}\right) \\ \left(5, id_2, \left\{ \text{gen} \mapsto (\text{gen}, \varepsilon) \right\}\right) \end{array} \right\}$$

Marker allocation

Marker are binary trees:

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

They are recursively calculated when resources are fetched.

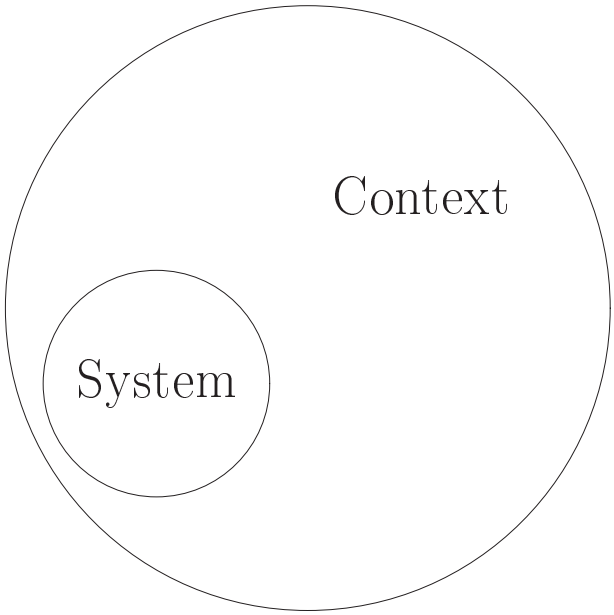
Coherence

Theorem: Standard semantics and non-standard semantics are strongly bisimilar.

The proof mainly relies on the consistence of marker allocation.

Context-free analysis

Analyzing interaction between a system and its unknown context.



The context may

- spy the system, by listening to message on unsafe channel names;
- spoil the system, by sending message via unsafe channel names.

Nasty context

$$\textbf{Context} := (\nu\ \text{unsafe})\ (\textbf{new} \mid \textbf{spy}_0 \mid \ldots \mid \textbf{spy}_n \mid \textbf{spoil}_0 \mid \ldots \mid \textbf{spoil}_n\ )$$

where

$$\textbf{new} := (*(\nu\ \mathit{channel})*\text{unsafe}![\mathit{channel}])$$

$$\textbf{spoil}_k := (*\text{unsafe}?[c]\text{unsafe}?[x_1]\ldots\text{unsafe}?[x_k]c![x_1,\ldots,x_k])$$

$$\textbf{spy}_k := (*\text{unsafe}![c]\text{c}?[x_1,\ldots,x_k]\ ((*\text{unsafe}![x_1]) \mid \ldots \mid (*\text{unsafe}![x_k])))$$

Abstraction

Abstract interpretation

Let $(\mathcal{C}, \rightarrow)$ be a transition system and $C_0 \in \mathcal{C}$,

$$\mathcal{S}(C_0) = \{C' \mid C_0 \rightarrow^* C'\} = lfp_{\emptyset}\mathbb{F}$$
$$\text{where } \mathbb{F} \ : \ X \mapsto \{C_0\} \cup \{C' | \exists C \in X, \ C \rightarrow C'\}$$

$$\gamma \ : \ \mathcal{D}^\sharp \rightarrow \wp(\mathcal{C}) \text{ with } \gamma(\bot_\sharp) = \emptyset$$
$$\mathbb{F}^\sharp \ : \ \mathcal{D}^\sharp \rightarrow \mathcal{D}^\sharp$$

such that

$$\forall d^\sharp \in \mathcal{D}^\sharp, \ [\mathbb{F} \circ \gamma](d^\sharp) \subseteq [\gamma \circ \mathbb{F}^\sharp](d^\sharp)$$

Theorem: $\boxed{\mathcal{S}(C_0) \subseteq \bigcup_{n \in \mathbb{N}} \gamma(\mathbb{F}^{\sharp^n}(\bot_\sharp))}$

Abstract semantics

We abstract

- for each sub-process, the set of markers it may be identified with;
- for each interaction (x, y), the set of pairs of markers (id_1, id_2) such that x free name of a thread whose marker was id_1, may be bound to the channel declared by the action $(\nu\ y)$ of a thread whose marker was id_2;
- for each channel name x, the set of markers id such that the channel name created by the action $(\nu\ x)$ of a thread whose marker was id may be unsafe.

Markers abstraction

We use both non-relational and relational domains.

- An automata-based non-relational domain is used in describing the general shape of markers.

  Widening operator is required to ensure the convergence of the analysis.

- a numerical relational domain is used to compare the number of occurrences of each pair of labels inside markers and inside pairs of markers.

Reduced product of these two domains provides accurate results.

Example

$$\mathcal{S} := (\nu\ \text{port})(\nu\ \text{gen})\ (\textbf{Server} \mid \textbf{Customer} \mid \text{gen}!^{6}[])$$

where

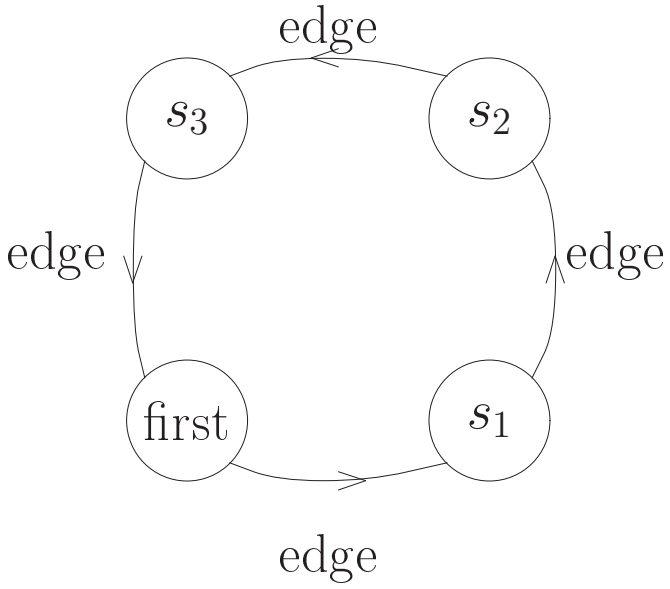
$$\textbf{Server} := *\text{port}?^{1}[info, add](add!^{2}[info])$$

$$\textbf{Customer} := *\text{gen}?^{3}[]\ (\nu\ data)\ (\nu\ email)\ (\text{port}!^{4}[data, email] \mid \text{gen}!^{5}[])$$

$$\begin{cases} (2, info, data) & \mapsto \left(\begin{array}{l} ((1,4)(3,5)^{*}(3,6), (3,5)^{*}(3,6)), \\ \{\sharp(3,5) = \sharp(3,5) \end{array} \right) \\ (4, data, data) & \mapsto \left(\begin{array}{l} ((3,5)^{*}(3,6), (3,5)^{*}(3,6)), \\ \{\sharp(3,5) = \sharp(3,5) \end{array} \right) \end{cases}$$

Example: a ring of processes

$$
\begin{array}{l}
(\nu\ \mathrm{make})(\nu\ \mathrm{edge})(\nu\ \mathrm{first}) \\
\quad (*\mathrm{make}?^{1}[\mathit{last}](\nu\ \mathit{next}) \\
\qquad\qquad\qquad\qquad (\mathrm{edge}!^{2}[\mathit{last},\mathit{next}] \mid \mathrm{make}!^{3}[\mathit{next}]) \\
\quad \mid *\mathrm{make}?^{4}[\mathit{last}](\mathrm{edge}!^{5}[\mathit{last},\mathrm{first}]) \\
\quad \mid \mathrm{make}!^{6}[\mathrm{first}])
\end{array}
$$



$$\sharp(1,3) + 1 = \sharp(1,3)$$

More example

$$
\begin{array}{l}
(\nu\ \text{make})(\nu\ \text{test}) \\
\quad (*\text{make}?^{1}[](\nu\ a)(\nu\ b) \\
\qquad\qquad\qquad (a?^{2}[]\, b?^{3}[]\, \cancel{\text{test}!^{4}[]} \\
\qquad\qquad\qquad \mid\ a?^{5}[]\, b!^{6}[] \\
\qquad\qquad\qquad \mid\ a!^{7}[]) \\
\quad \mid\ \text{make}!^{8}[])
\end{array}
$$

Context-free analysis allows to prove that subprocess $\text{test}!^{4}[]$ is unreachable.

Conclusion

- Our framework allows to infer a sound non-uniform description of mobile systems in the π-calculus.
- Context free analysis has many applications:
  - modular analysis;
  - more precise results;
  - confidentiality analysis in hostile context.
- Our methodology can be adapted to other formalism(*mobile ambients*).

Future Works

- occurrences counting analysis

  (exhaustion of resources, mutual exclusion)
- automatic detection of deadlocks