Capacity-Bounded Computational Ambients

FOSAD 2002, Bertinoro 25.09.02

Vladimiro Sassone

University of Sussex,

A Global Computing Scenario

Global Computing refers to computation via the sharing of a seamless, distributed, open-ended network of bounded resource by agents of all sort (possibly malicious), acting with partial knowledge and no central coordination.

A Global Computing Scenario

Global Computing refers to computation via the sharing of a seamless, distributed, open-ended network of bounded resource by agents of all sort (possibly malicious), acting with partial knowledge and no central coordination.

AmI: Ambient Intelligence: Some of the technologies just behind the corner

- *Personal Area Networks (PAN)*
- *Personal Communication Device (P-Com)*
- *Automated Debiting Systems (ADS)*
- *Vehicle Area Networks (VAN)*
- *Digital Me (D-Me)*
- *. . .*

Devices that are carried along *and* exit *and* enter *domains.*

A Global Computing Scenario

Global Computing refers to computation via the sharing of a seamless, distributed, open-ended network of bounded resource by agents of all sort (possibly malicious), acting with partial knowledge and no central coordination.

AmI: Ambient Intelligence: Some of the technologies just behind the corner

- *Personal Area Networks (PAN)*
- *Personal Communication Device (P-Com)*
- *Automated Debiting Systems (ADS)*
- *Vehicle Area Networks (VAN)*
- *Digital Me (D-Me)*
- *. . .*

Devices that are carried along *and* exit *and* enter *domains.*

Some of the aspects involved:

- *localities*
- *diversity*
- *open endedness*
- *no a priori trustworthy authority*
- *partial knowledge. . .*
- *mobility and migration*
- *communication*
- *no central control*
- *malicious entities*
- *. . . and its acquisition*

Global Computing Requirements

Requirements:

- Dynamic Learning and Checking about Environment and Peers
- Trust Formation and Management
- Location Awareness
- Security: Authentication, Privacy, Non Repudiation
- Policies of Access Control and their Enforcement
- Negotiation of Access, Access Rights, Resource Acquisition
- Protection of Resource Bounds
- and more...

Typical Devices:

Today: Smart Cards, Embedded devs (e.g. in cars), Mobile phones, PDAs, Sat navigators, ...

Tomorrow: PAN, VAN, D-ME, P-COM, ...

Global Computing Requirements

Requirements:

- Dynamic Learning and Checking about Environment and Peers
- Trust Formation and Management
- Location Awareness
- Security: Authentication, Privacy, Non Repudiation
- Policies of Access Control and their Enforcement
- Negotiation of Access, Access Rights, Resource Acquisition
- Protection of Resource Bounds
- and more...

Typical Devices:

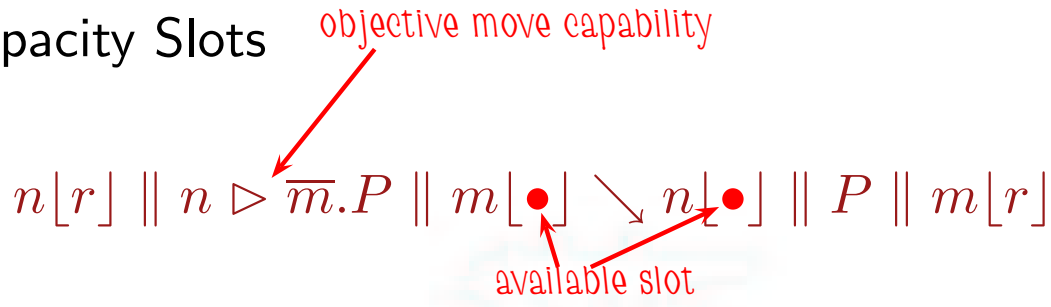
Today: Smart Cards, Embedded devs (e.g. in cars), Mobile phones, PDAs, Sat navigators, ...

Tomorrow: PAN, VAN, D-ME, P-COM, ...

In this lecture we focus on Capacity Bounds Awareness.

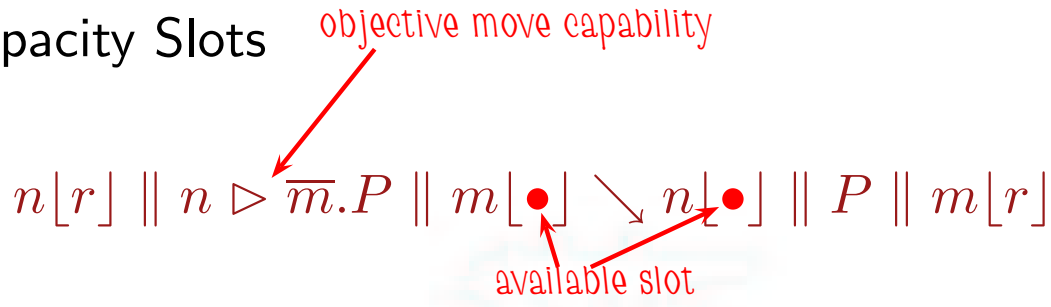
Roadmap

- Part I – Objective Mobility: MR
  - Objective Mobility
  - Bounded Capacity Slots

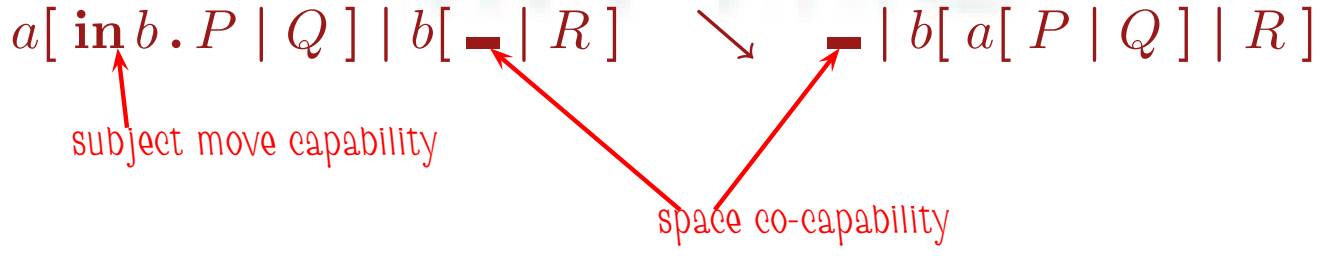


Roadmap

- Part I – Objective Mobility: MR
  - Objective Mobility
  - Bounded Capacity Slots



- Part II – Subjective Mobility: BoCa
  - Subjective Mobility
  - Bounded Capacity Ambients
  - Space as a linear co-capability. Finer Control of Capacity.



The elements of the calculus

- Empty slot: $n[\bullet]$;

A Calculus of Mobile Resources

The elements of the calculus

- Empty slot: $n\lfloor\bullet\rfloor$;
- Slot containing a resource $n\lfloor r\rfloor$; unitary slot capacity:
  - $n\lfloor\bullet\rfloor$ may receive a resource from the context: $n\lfloor\bullet\rfloor \rightsquigarrow n\lfloor r\rfloor$;
  - $n\lfloor r\rfloor$ can only emit r to the context: $n\lfloor r\rfloor \rightsquigarrow n\lfloor\bullet\rfloor$;

A Calculus of Mobile Resources

The elements of the calculus

- Empty slot: $n\lfloor\bullet\rfloor$;
- Slot containing a resource $n\lfloor r\rfloor$; unitary slot capacity:
  - $n\lfloor\bullet\rfloor$ may receive a resource from the context: $n\lfloor\bullet\rfloor \rightsquigarrow n\lfloor r\rfloor$;
  - $n\lfloor r\rfloor$ can only emit r to the context: $n\lfloor r\rfloor \rightsquigarrow n\lfloor\bullet\rfloor$;
- Objective movement: $n\lfloor r\rfloor \parallel n \rhd \overline{m}.p \parallel m\lfloor\bullet\rfloor \searrow n\lfloor\bullet\rfloor \parallel p \parallel m\lfloor r\rfloor$

A Calculus of Mobile Resources

The elements of the calculus

- Empty slot: $n\lfloor\bullet\rfloor$;
- Slot containing a resource $n\lfloor r\rfloor$; unitary slot capacity:
  - $n\lfloor\bullet\rfloor$ may receive a resource from the context: $n\lfloor\bullet\rfloor \rightsquigarrow n\lfloor r\rfloor$;
  - $n\lfloor r\rfloor$ can only emit r to the context: $n\lfloor r\rfloor \rightsquigarrow n\lfloor\bullet\rfloor$;
- Objective movement: $n\lfloor r\rfloor \parallel n \rhd \overline{m}.p \parallel m\lfloor\bullet\rfloor \searrow n\lfloor\bullet\rfloor \parallel p \parallel m\lfloor r\rfloor$
- Synchronisation:
  - local: $\alpha.p \parallel \overline{\alpha}.q \searrow p \parallel q$
  - with resources: $n\alpha.p \parallel n\lfloor\overline{\alpha}.q\rfloor \searrow p \parallel n\lfloor q\rfloor$

A Calculus of Mobile Resources

The elements of the calculus

- Empty slot: $n\lfloor\bullet\rfloor$;
- Slot containing a resource $n\lfloor r\rfloor$; unitary slot capacity:
  - $n\lfloor\bullet\rfloor$ may receive a resource from the context: $n\lfloor\bullet\rfloor \rightsquigarrow n\lfloor r\rfloor$;
  - $n\lfloor r\rfloor$ can only emit r to the context: $n\lfloor r\rfloor \rightsquigarrow n\lfloor\bullet\rfloor$;
- Objective movement: $n\lfloor r\rfloor \parallel n \rhd \overline{m}.p \parallel m\lfloor\bullet\rfloor \searrow n\lfloor\bullet\rfloor \parallel p \parallel m\lfloor r\rfloor$
- Synchronisation:
  - local: $\alpha.p \parallel \overline{\alpha}.q \searrow p \parallel q$
  - with resources: $n\alpha.p \parallel n\lfloor\overline{\alpha}.q\rfloor \searrow p \parallel n\lfloor q\rfloor$
- Slot deletion: $n\lfloor\!\lfloor r \rfloor\!\rfloor_m \parallel \natural m.q \searrow q$

A Calculus of Mobile Resources

The elements of the calculus

- Empty slot: $n\lfloor\bullet\rfloor$;
- Slot containing a resource $n\lfloor r\rfloor$; unitary slot capacity:
  - $n\lfloor\bullet\rfloor$ may receive a resource from the context: $n\lfloor\bullet\rfloor \rightsquigarrow n\lfloor r\rfloor$;
  - $n\lfloor r\rfloor$ can only emit r to the context: $n\lfloor r\rfloor \rightsquigarrow n\lfloor\bullet\rfloor$;
- Objective movement: $n\lfloor r\rfloor \parallel n \rhd \overline{m}.p \parallel m\lfloor\bullet\rfloor \searrow n\lfloor\bullet\rfloor \parallel p \parallel m\lfloor r\rfloor$
- Synchronisation:
  - local: $\alpha.p \parallel \overline{\alpha}.q \searrow p \parallel q$
  - with resources: $n\alpha.p \parallel n\lfloor\overline{\alpha}.q\rfloor \searrow p \parallel n\lfloor q\rfloor$
- Slot deletion: $n\lfloor\, r\,\rfloor_m \parallel \natural m.q \searrow q$
- Resource hiding $n\lfloor r\rfloor \parallel (k)(n \rhd \overline{k}.p \parallel k\lfloor\bullet\rfloor) \searrow n\lfloor\bullet\rfloor \parallel (k)(p \parallel k\lfloor r\rfloor)$

A Calculus of Mobile Resources

The elements of the calculus

- Empty slot: $n\lfloor\bullet\rfloor$;
- Slot containing a resource $n\lfloor r\rfloor$; unitary slot capacity:
  - $n\lfloor\bullet\rfloor$ may receive a resource from the context: $n\lfloor\bullet\rfloor \rightsquigarrow n\lfloor r\rfloor$;
  - $n\lfloor r\rfloor$ can only emit r to the context: $n\lfloor r\rfloor \rightsquigarrow n\lfloor\bullet\rfloor$;
- Objective movement: $n\lfloor r\rfloor \parallel n \rhd \overline{m}.p \parallel m\lfloor\bullet\rfloor \searrow n\lfloor\bullet\rfloor \parallel p \parallel m\lfloor r\rfloor$
- Synchronisation:
  - local: $\alpha.p \parallel \overline{\alpha}.q \searrow p \parallel q$
  - with resources: $n\alpha.p \parallel n\lfloor\overline{\alpha}.q\rfloor \searrow p \parallel n\lfloor q\rfloor$
- Slot deletion: $n\lfloor r \rfloor_m \parallel \natural m.q \searrow q$
- Resource hiding $n\lfloor r\rfloor \parallel (k)(n \rhd \overline{k}.p \parallel k\lfloor\bullet\rfloor) \searrow n\lfloor\bullet\rfloor \parallel (k)(p \parallel k\lfloor r\rfloor)$
- Deep nesting of resources: $n_0\lfloor\cdots n_k\lfloor r\rfloor\cdots\rfloor$

MR: Syntax

- Names $\mathcal{N}$ and co-names $\overline{\mathcal{N}}$: $\alpha \in \mathcal{N} \cup \overline{\mathcal{N}}$.
- Direction paths: $\delta \in \mathcal{N}^+$, $\gamma \in \mathcal{N}^*$.
- Action prefixes: $\lambda ::= \gamma\alpha \mid \delta \rhd \overline{\delta'} \mid \natural\gamma n$
- Processes: $\mathcal{P}$:

$$p, q ::= 0 \mid \lambda.p \mid p \parallel p \mid (n)p \mid !p \mid n\lfloor r \rfloor_m$$
$$r ::= \bullet \mid p$$

MR: Structural Congruence

- A structural congruence is used to quotient terms to suitably abstract entities, so as to dispense with boring details (e.g. associativity of $\|$).

- $\equiv$ is the least congruence on $\mathcal{P}$ satisfying alpha-conversion and

$$p \parallel 0 \equiv p$$

$$p \parallel q \equiv q \parallel p$$

$$(p \parallel q) \parallel s \equiv p \parallel (q \parallel s)$$

$$(k)0 \equiv 0$$

$$(k)p \parallel q \equiv (k)(p \parallel q), \qquad \text{if } n \notin fn(q)$$

$$(k)n\lfloor r \rfloor_m \equiv n\lfloor (k)p \rfloor_m, \qquad \text{if } k \notin \{n, m\}$$

$$!p \equiv !p \parallel p$$

What is a Reduction Systems

- A reduction system over a signature Σ is a relation $\searrow \subseteq T_\Sigma \times T_\Sigma$, T_Σ is the set of terms Σ.

- Reduction systems are often presented parametrically.

Contexts: terms with variables: $\mathscr{C}[x_1, \ldots, x_n]$

Reduction rules: set $\mathscr{R}$ of parametric rewriting rules:
$\mathscr{C}[x_1, \ldots, x_n] \searrow \mathscr{D}[x_1, \ldots, x_n]$.

Evaluation Contexts: chosen set $\mathscr{E}$ of single-variable contexts.

$$\frac{\mathscr{C}[x_1, \ldots, x_n] \searrow \mathscr{D}[x_1, \ldots, x_n] \in \mathscr{R} \qquad \mathscr{E} \text{ evaluation context}}{\mathscr{E}[\mathscr{C}[t_1, \ldots, t_n]] \searrow \mathscr{E}[\mathscr{D}[t_1, \ldots, t_n]]}$$

MR: Reduction Semantics

- Direction path contexts $\mathscr{C}_\gamma$ and $\mathscr{D}_\delta$ by

$$\mathscr{C}_\epsilon ::= (-) \qquad \mathscr{C}_{n\gamma} ::= n\lfloor \mathscr{C}_\gamma \parallel p \rfloor_m$$

$$\mathscr{D}_{\gamma n} ::= \mathscr{C}_\gamma(n\lfloor (-) \rfloor_m)x$$

- $\searrow$ is the least relation closed under $\equiv$ and under evaluation contexts, $\mathscr{E}$ such that

$$\begin{array}{rcl}
\gamma\delta_1 \rhd \overline{\gamma\delta_2}.p \parallel \mathscr{C}_\gamma\big(\mathscr{D}_{\delta_1}(r) \parallel \mathscr{D}_{\delta_2}(\bullet)\big) & \searrow & p \parallel \mathscr{C}_\gamma\big(\mathscr{D}_{\delta_1}(\bullet) \parallel \mathscr{D}_{\delta_2}(r)\big) \\
\gamma\alpha.p \parallel \mathscr{C}_\gamma(\overline{\alpha}.q) & \searrow & p \parallel \mathscr{C}_\gamma(q) \\
\natural\gamma m.p \parallel \mathscr{C}_\gamma(n\lfloor r \rfloor_m) & \searrow & p \parallel \mathscr{C}_\gamma(0)
\end{array}$$

where $\mathscr{E} ::= (-) \mid n\lfloor \mathscr{E} \rfloor_m \mid E \parallel p \mid (n)E$.

Reduction Barbed Congruence

- Barbs:

$$p{\downarrow}n \quad \text{if} \quad p \equiv (\tilde{n})(\alpha.p' \parallel q), \quad \text{where} \quad \alpha \in \{n, \overline{n}\} \text{ and } n \notin \tilde{n}.$$

Reduction Barbed Congruence

- Barbs:

$$p{\downarrow}n \quad \text{if} \quad p \equiv (\tilde{n})(\alpha.p' \parallel q), \quad \text{where} \quad \alpha \in \{n, \overline{n}\} \text{ and } n \notin \tilde{n}.$$

- Barbed bisimulation congruence:

The largest congruence $\sim_b$ such that if $p \sim_b q$ then

  - $p \searrow p'$ implies $\exists q \searrow q'$ such that $p' \sim_b q'$ (reduction closed)
  - $p{\downarrow}n$ iff $q{\downarrow}n$ (barb-preserving)

Reduction Barbed Congruence

- *Barbs*:

$$p{\downarrow}n \quad \text{if} \quad p \equiv (\tilde{n})(\alpha.p' \parallel q), \quad \text{where} \quad \alpha \in \{n, \overline{n}\} \text{ and } n \notin \tilde{n}.$$

- *Barbed bisimulation congruence*:

  The largest congruence $\sim_b$ such that if $p \sim_b q$ then
  - $p \searrow p'$ implies $\exists q \searrow q'$ such that $p' \sim_b q'$ (reduction closed)
  - $p{\downarrow}n$ iff $q{\downarrow}n$ (barb-preserving)

- *Note*: To prove that $p \sim_b q$ one must prove that they (and all their reducts) produce the matching observations (barbs) in all contexts (congruence).

Example

Resources cannot be copied.

A model of a pre-paid cash card (the resource b) in the slot a of a vending machine that delivers a cup of coffee (action c) for each cash card of the right kind, b, inserted in a.

$$(b)\big(\, a\lfloor b\rfloor \;\parallel !a\bar{b}.c\,\big) \sim_b (b)\big(\, a\lfloor b\rfloor \;\parallel\; a\bar{b}.c\,\big)$$

If there exists only *one* card of the 'right' type, then there will ever be only *one* cup of coffee: the cash card cannot be copied.

Example

Resources cannot be copied.

A model of a pre-paid cash card (the resource b) in the slot a of a vending machine that delivers a cup of coffee (action c) for each cash card of the right kind, b, inserted in a.

$$(b)\big(\, a\lfloor b\rfloor \;\parallel !a\bar{b}.c\,\big) \sim_b (b)\big(\, a\lfloor b\rfloor \;\parallel\; a\bar{b}.c\,\big)$$

If there exists only *one* card of the 'right' type, then there will ever be only *one* cup of coffee: the cash card cannot be copied.

Exercise:

Use the operational definition to convince yourself of the $\sim_b$-equation above.

Scope and Mobility

The interplay of upward and downward moves and scope is a major challenge.

$$\mathscr{C}_1 \triangleq c\lfloor(-)\rfloor \parallel a, \qquad \mathscr{C}_2 \triangleq d\lfloor(-)\rfloor \parallel d\overline{a}\,.\,b$$

$$p \triangleq (a)\mathscr{C}_1(\mathscr{C}_2(\bullet)) = (a)\big(\,c\lfloor d\lfloor\bullet\rfloor \;\parallel\; d\overline{a}\,.\,b\rfloor \;\parallel\; a\,\big).$$

Scope and Mobility

The interplay of upward and downward moves and scope is a major challenge.

$$\mathscr{C}_1 \triangleq c\lfloor(-)\rfloor \parallel a, \qquad \mathscr{C}_2 \triangleq d\lfloor(-)\rfloor \parallel d\overline{a}\,.\,b$$

$$p \triangleq (a)\mathscr{C}_1(\mathscr{C}_2(\bullet)) = (a)\big(\,c\lfloor d\lfloor\bullet\rfloor \parallel d\overline{a}\,.\,b\rfloor \parallel a\,\big).$$

It seems that no resource can be inserted into the empty slot d to synchronise with the $d\overline{a}$-action; similarly, no process can ever synchronise with the a action at top-level. It then follow that p behaves like $q \triangleq (a)(c\lfloor d\lfloor\bullet\rfloor\rfloor)$.

Yet, this is not the case. Under a suitable context, it is possible for the process a to change its role from being the parent of $d\overline{a}\,.\,b$ to being its child in the slot named d.

Scope and Mobility

The interplay of upward and downward moves and scope is a major challenge.

$$\mathscr{C}_1 \triangleq c\lfloor(-)\rfloor \parallel a, \qquad \mathscr{C}_2 \triangleq d\lfloor(-)\rfloor \parallel d\overline{a}\,.\,b$$

$$p \triangleq (a)\mathscr{C}_1(\mathscr{C}_2(\bullet)) = (a)\big(\,c\lfloor d\lfloor\bullet\rfloor \parallel d\overline{a}\,.\,b\rfloor \parallel a\,\big).$$

Suppose in fact that p and q are inserted into the context

$$\mathscr{C} = x\lfloor(-)\rfloor \parallel y\lfloor\bullet\rfloor \parallel xc \rhd \overline{y}\,.\,x \rhd \overline{yd}.$$

Then $\mathscr{C}(p)$ reduces (in two steps) to

$$(a)\big(\,x\lfloor\bullet\rfloor \parallel y\lfloor\,\mathscr{C}_2(\mathscr{C}_1(\bullet))\,\rfloor\big) = (a)\big(\,x\lfloor\bullet\rfloor \parallel y\lfloor d\lfloor c\lfloor\bullet\rfloor \parallel a\rfloor \parallel d\overline{a}\,.\,b\rfloor\big),$$

where $\mathscr{C}_1$ and $\mathscr{C}_2$ have swapped place. Now the b-action may be unleashed upon synchronisation on a.

Digital Signature Card

Enc_k (Dec_k) encrypts (decrypts) resources received in its *in*-buffer with key k, and returns the encrypted resource via its *out*-buffer.

$$Enc_k \triangleq !(reg)(\, in \rhd \overline{reg\ k}\,.\, reg \rhd \overline{out} \parallel reg\lfloor k\lfloor \bullet \rfloor \rfloor) \parallel in\lfloor \bullet \rfloor \parallel out\lfloor \bullet \rfloor$$

$$Dec_k \triangleq !(reg)(in \rhd \overline{reg}\,.\, reg\ k \rhd \overline{out}\,. \parallel reg\lfloor \bullet \rfloor) \parallel in\lfloor \bullet \rfloor \parallel out\lfloor \bullet \rfloor$$

Digital Signature Card

Enc_k (Dec_k) encrypts (decrypts) resources received in its *in*-buffer with key k, and returns the encrypted resource via its *out*-buffer.

$$Enc_k \triangleq !(reg)(\, in \rhd \overline{reg\ k}\,.\, reg \rhd \overline{out} \parallel reg\lfloor k\lfloor \bullet \rfloor \rfloor) \parallel in\lfloor \bullet \rfloor \parallel out\lfloor \bullet \rfloor$$

$$Dec_k \triangleq !(reg)(in \rhd \overline{reg}\,.\, reg\ k \rhd \overline{out}\,. \parallel reg\lfloor \bullet \rfloor) \parallel in\lfloor \bullet \rfloor \parallel out\lfloor \bullet \rfloor$$

If k is a shared secret between Alice and Bob, then Alice can send messages secretly to Bob.

$$Alice_{k,M} \triangleq (m)(a)\big(\, a\lfloor Enc_k \rfloor \parallel m\lfloor M \rfloor \parallel m \rhd \overline{a\ in}\,.\, a\ out \rhd \overline{network}\,\big)$$

$$Bob_k \triangleq (m)(b)(b\lfloor Dec_k \rfloor \parallel m\lfloor \bullet \rfloor \parallel network \rhd \overline{b\ in}\,.\, b\ out \rhd \overline{m})$$

$$SecretCom_M \triangleq (k)\big(\, Alice_{k,M} \parallel Bob_k\,\big) \parallel network\lfloor \bullet \rfloor$$

Digital Signature Card

Enc_k (Dec_k) encrypts (decrypts) resources received in its *in*-buffer with key k, and returns the encrypted resource via its *out*-buffer.

$$Enc_k \triangleq !(reg)(\, in \rhd \overline{reg\ k}\,.\, reg \rhd \overline{out} \parallel reg\lfloor k\lfloor \bullet \rfloor \rfloor) \parallel in\lfloor \bullet \rfloor \parallel out\lfloor \bullet \rfloor$$

$$Dec_k \triangleq !(reg)(in \rhd \overline{reg}\,.\, reg\ k \rhd \overline{out}\,. \parallel reg\lfloor \bullet \rfloor) \parallel in\lfloor \bullet \rfloor \parallel out\lfloor \bullet \rfloor$$

If k is a shared secret between Alice and Bob, then Alice can send messages secretly to Bob.

$$Alice_{k,M} \triangleq (m)(a)\big(\, a\lfloor Enc_k \rfloor \parallel m\lfloor M \rfloor \parallel m \rhd \overline{a\ in}\,.\, a\ out \rhd \overline{network}\,\big)$$

$$Bob_k \triangleq (m)(b)(b\lfloor Dec_k \rfloor \parallel m\lfloor \bullet \rfloor \parallel network \rhd \overline{b\ in}\,.\, b\ out \rhd \overline{m})$$

$$SecretCom_M \triangleq (k)\big(\, Alice_{k,M} \parallel Bob_k \,\big) \parallel network\lfloor \bullet \rfloor$$

For all messages of the form $M = a_1\,.\,a_2\,.\,\ldots\,.\,a_i$ and $M' = a'_1\,.\,a'_2\,.\,\ldots\,.\,a'_j$:

$$SecretCom_M \sim_b SecretCom_{M'}.$$

Digital Signature Card

Enc_k (Dec_k) encrypts (decrypts) resources received in its *in*-buffer with key k, and returns the encrypted resource via its *out*-buffer.

$$Enc_k \triangleq !(reg)(\, in \rhd \overline{reg\ k}\,.\, reg \rhd \overline{out} \parallel reg\lfloor k\lfloor \bullet \rfloor \rfloor) \parallel in\lfloor \bullet \rfloor \parallel out\lfloor \bullet \rfloor$$

$$Dec_k \triangleq !(reg)(in \rhd \overline{reg}\,.\, reg\ k \rhd \overline{out}\,. \parallel reg\lfloor \bullet \rfloor) \parallel in\lfloor \bullet \rfloor \parallel out\lfloor \bullet \rfloor$$

If k is a shared secret between Alice and Bob, then Alice can send messages secretly to Bob.

$$Alice_{k,M} \triangleq (m)(a)\big(\, a\lfloor Enc_k \rfloor \parallel m\lfloor M \rfloor \parallel m \rhd \overline{a\ in}\,.\, a\ out \rhd \overline{network}\,\big)$$

$$Bob_k \triangleq (m)(b)(b\lfloor Dec_k \rfloor \parallel m\lfloor \bullet \rfloor \parallel network \rhd \overline{b\ in}\,.\, b\ out \rhd \overline{m})$$

$$SecretCom_M \triangleq (k)\big(\, Alice_{k,M} \parallel Bob_k \,\big) \parallel network\lfloor \bullet \rfloor$$

Digital signature card which generates the key and exports the decryption resource (as many times as needed) but keeps the encryption resource private.

$$SignatureCard \triangleq (k)\big(\, !export\lfloor Dec_k \rfloor \parallel Enc_k \,\big)$$

What is a Labelled Transition System?

- Rather than describing the internal behaviour of a system (reductions) it describe the interactions this is willing to offer to the surrounding enviroment.
These are characterised and described using label transitions, where a transition indicates an activitiy and a label classifies it.
For instance $\text{client} - \textit{`insert coin'} \longrightarrow \text{client}'$. Or perhaps,
$\text{machine} - \textit{`delivers candy'} \longrightarrow \text{machine}'$.

- This yield a compositional semantics, as e.g.:

$$\frac{\text{client} - \textit{`insert coin'} \longrightarrow \text{client}' \quad \text{machine} - \textit{`delivers candy'} \longrightarrow \text{machine}'}{\text{client} \mid \text{machine} - \textit{`jum'} \longrightarrow \text{client}' \mid \text{machine}'}$$

- Label transition systems admit proof techniques (LTS bisimulation), verification of logic formulas (model-checking), ...

MR: A Compositional Operational Semantics

The labelled transition semantics consists of CCS-like rules plus rules for:

- Nested resources
- Mobility – a three party interaction: move + exit + enter
- Scope extension
- Slot deletion

MR: A Compositional Operational Semantics

The labelled transition semantics consists of CCS-like rules plus rules for:

- Nested resources
- Mobility – a three party interaction: move + exit + enter
- Scope extension
- Slot deletion

(prefix)

$$\frac{}{\lambda\,.\,p \xrightarrow{\lambda} p}$$

(rest)

$$\frac{p \xrightarrow{\pi} p'}{(n)p \xrightarrow{\pi} (n)p'}\; n \notin fn(\pi) \cup bn(\pi)$$

(rep)

$$\frac{p \parallel !p \xrightarrow{\pi} p'}{!p \xrightarrow{\pi} p'}$$

(sync)

$$\frac{p_1 \xrightarrow{(\tilde{n})\overline{\pi}} p_1' \quad p_2 \xrightarrow{\pi} p_2'}{p_1 \parallel p_2 \xrightarrow{\tau} (\tilde{n})(p_1' \parallel p_2')}\; fn(p_2) \cap \tilde{n} = \emptyset$$

(sym)

$$\frac{p \parallel q \xrightarrow{\pi} p' \parallel q}{q \parallel p \xrightarrow{\pi} q \parallel p'}$$

(par)

$$\frac{p \xrightarrow{\pi} p'}{p \parallel q \xrightarrow{\pi} p' \parallel q}\; fn(q) \cap bn(\pi) = \emptyset$$

Labelled Transition Semantics – Nesting

Labels of the form $\overline{\delta}\alpha$ capture path communication by synchronising with directed actions of the form $\delta\alpha$ which appear as prefixes in the calculus.

Labelled Transition Semantics – Nesting

Labels of the form $\overline{\delta}\alpha$ capture path communication by synchronising with directed actions of the form $\delta\alpha$ which appear as prefixes in the calculus.

For instance: $n\lfloor a.p \rfloor \xrightarrow{\overline{n}a} n\lfloor p \rfloor$

Labelled Transition Semantics – Nesting

Labels of the form $\overline{\delta}\alpha$ capture path communication by synchronising with directed actions of the form $\delta\alpha$ which appear as prefixes in the calculus.

$$\text{(nesting)}\quad \frac{p \xrightarrow{\pi} p'}{n\lfloor p \rfloor \xrightarrow{n\cdot(\pi)} n\lfloor p' \rfloor}$$

where $\cdot$ 'distributes' $\overline{n}$ over (a potentially complex) π.

Labelled Transition Semantics – Nesting

Labels of the form $\overline{\delta}\alpha$ capture path communication by synchronising with directed actions of the form $\delta\alpha$ which appear as prefixes in the calculus.

$$\text{(nesting)}\quad \frac{p \xrightarrow{\pi} p'}{n\lfloor p \rfloor \xrightarrow{n\cdot(\pi)} n\lfloor p' \rfloor}$$

where $\cdot$ 'distributes' $\overline{n}$ over (a potentially complex) π.

$$\frac{\dfrac{\overline{\alpha}.q \xrightarrow{\overline{\alpha}} q}{m\lfloor \overline{\alpha}.q \rfloor \xrightarrow{\overline{m\alpha}} m\lfloor q \rfloor} \quad \dfrac{}{m\alpha.p' \xrightarrow{m\alpha} p'}}{m\lfloor \overline{\alpha}.q \rfloor \parallel m\alpha.p' \xrightarrow{\tau} m\lfloor q \rfloor \parallel p'}}{n\lfloor m\lfloor \overline{\alpha}.q \rfloor \parallel m\alpha.p' \rfloor \xrightarrow{\tau} n\lfloor m\lfloor q \rfloor \parallel p' \rfloor}$$

where $m \cdot (\overline{\alpha}) = \overline{m\alpha}$ and $m \cdot (\tau) = \tau$

Labelled Transition Semantics – HO Labels

- Modelling the three-party interaction required for the movement of resources needs higher-order labels.

$$\overline{\delta}\triangleright\langle p\rangle \quad (p \text{ exits from } \delta) \qquad \text{and} \qquad (p)\triangleright\delta \quad (p \text{ enters in } \delta).$$

- The corresponding co-labels are $\delta\triangleright(p)$ and $\langle p\rangle\triangleright\overline{\delta}$.

Labelled Transition Semantics – HO Labels

- Modelling the three-party interaction required for the movement of resources needs higher-order labels.

$$\overline{\delta}\triangleright\langle p\rangle \quad (p \text{ exits from } \delta) \qquad \text{and} \qquad (p)\triangleright\delta \quad (p \text{ enters in } \delta).$$

- The corresponding co-labels are $\delta\triangleright(p)$ and $\langle p\rangle\triangleright\overline{\delta}$.

For instance

$$n\lfloor p\rfloor \xrightarrow{\overline{n}\triangleright\langle p\rangle} n\lfloor\bullet\rfloor \quad (\text{exit}), \qquad m\lfloor\bullet\rfloor \xrightarrow{(p)\triangleright m} m\lfloor p\rfloor \quad (\text{enter}).$$

which may synchronise

$$n\lfloor p\rfloor \parallel m\lfloor\bullet\rfloor \xrightarrow{\overline{n}\triangleright m} n\lfloor\bullet\rfloor \parallel m\lfloor p\rfloor \quad (\text{co-move})$$

Labels of the form $\overline{\delta_1}\triangleright\delta_2$ are the co-actions of the (move) action $\delta_1 \triangleright \overline{\delta_2}$

$$n\lfloor p\rfloor \parallel m\lfloor\bullet\rfloor \parallel n \triangleright \overline{m}\,.\,q \xrightarrow{\tau} n\lfloor\bullet\rfloor \parallel m\lfloor p\rfloor \parallel q$$

Labelled Transition Semantics – HO Labels

- Modelling the three-party interaction required for the movement of resources needs higher-order labels.

$$\overline{\delta}\triangleright\langle p\rangle \quad (p \text{ exits from } \delta) \qquad \text{and} \qquad (p)\triangleright\delta \quad (p \text{ enters in } \delta).$$

- The corresponding co-labels are $\delta\triangleright(p)$ and $\langle p\rangle\triangleright\overline{\delta}$.
- Labels of the form $\overline{\delta_1}\triangleright\delta_2$ are the co-actions of the $(\mathtt{move})$ action $\delta_1 \triangleright \overline{\delta_2}$

Labelled Transition Semantics – HO Labels

- Modelling the three-party interaction required for the movement of resources needs higher-order labels.

$$\overline{\delta}\triangleright\langle p\rangle \quad (p \text{ exits from } \delta) \qquad \text{and} \qquad (p)\triangleright\delta \quad (p \text{ enters in } \delta).$$

- The corresponding co-labels are $\delta\triangleright(p)$ and $\langle p\rangle\triangleright\overline{\delta}$.
- Labels of the form $\overline{\delta_1}\triangleright\delta_2$ are the co-actions of the (move) action $\delta_1 \triangleright \overline{\delta_2}$

Also, (exit) and (move) transitions may

$$n\lfloor p\rfloor \parallel n \triangleright \overline{m}\,.\,q \xrightarrow{\langle p\rangle\triangleright\overline{m}} n\lfloor\bullet\rfloor \parallel q \qquad \text{(give)}$$

ready for a (enter) transition. Same for (enter) and (move)

$$m\lfloor\bullet\rfloor \parallel n \triangleright \overline{m}\,.\,q \xrightarrow{n\triangleright(p)} m\lfloor p\rfloor \parallel q \qquad \text{(take)},$$

ready to synchronise with the dual (exit) transition.

Labelled Transition Semantics – HO Labels

- Modelling the three-party interaction required for the movement of resources needs higher-order labels.

$$\overline{\delta}\triangleright\langle p\rangle \quad (p \text{ exits from } \delta) \qquad \text{and} \qquad (p)\triangleright\delta \quad (p \text{ enters in } \delta).$$

- The corresponding co-labels are $\delta\triangleright(p)$ and $\langle p\rangle\triangleright\overline{\delta}$.
- Labels of the form $\overline{\delta_1}\triangleright\delta_2$ are the co-actions of the $(\mathtt{move})$ action $\delta_1 \triangleright \overline{\delta_2}$
- Summing up

$\delta_1 \triangleright \overline{\delta_2}$	coalesces with	$\overline{\delta_1}\triangleright\langle p\rangle$	yielding	$\langle p\rangle\triangleright\overline{\delta_2}$,
$\delta_1 \triangleright \overline{\delta_2}$	coalesces with	$(p)\triangleright\delta_2$	yielding	$\delta_1\triangleright(p)$,
$\overline{\delta_1}\triangleright\langle p\rangle$	coalesces with	$(p)\triangleright\delta_2$	yielding	$\overline{\delta_1}\triangleright\delta_2$.

Labelled Transition Semantics – Mobility (I)

(exit)

$$\frac{}{n\lfloor p \rfloor_m \xrightarrow{\overline{n}\rhd\langle p\rangle} n\lfloor \bullet \rfloor_m}$$

(enter)

$$\frac{}{n\lfloor \bullet \rfloor_m \xrightarrow{(p)\rhd n} n\lfloor p \rfloor_m}$$

(co-move)

$$\frac{p_1 \xrightarrow{\overline{\delta_1}\rhd\langle q\rangle} p_1' \quad p_2 \xrightarrow{(q)\rhd\delta_2} p_2'}{p_1 \parallel p_2 \xrightarrow{\overline{\delta_1}\rhd\delta_2} p_1' \parallel p_2'}$$

Labelled Transition Semantics – Mobility (I)

(exit)

$$n\lfloor p \rfloor_m \xrightarrow{\overline{n}\rhd\langle p\rangle} n\lfloor \bullet \rfloor_m$$

(enter)

$$n\lfloor \bullet \rfloor_m \xrightarrow{(p)\rhd n} n\lfloor p \rfloor_m$$

(co-move)

$$\frac{p_1 \xrightarrow{\overline{\delta_1}\rhd\langle q\rangle} p_1' \quad p_2 \xrightarrow{(q)\rhd\delta_2} p_2'}{p_1 \parallel p_2 \xrightarrow{\overline{\delta_1}\rhd\delta_2} p_1' \parallel p_2'}$$

$$\frac{\dfrac{}{k\lfloor q\rfloor \xrightarrow{\overline{k}\rhd\langle q\rangle} k\lfloor\bullet\rfloor}}{n\lfloor k\lfloor q\rfloor\rfloor \xrightarrow{\overline{nk}\rhd\langle q\rangle} n\lfloor k\lfloor\bullet\rfloor\rfloor} \qquad \frac{}{m\lfloor\bullet\rfloor \xrightarrow{(q)\rhd m} m\lfloor q\rfloor}$$

$$\frac{\dfrac{}{nk \rhd \overline{m}.p \xrightarrow{nk\rhd\overline{m}} p} \qquad n\lfloor k\lfloor q\rfloor\rfloor \parallel m\lfloor\bullet\rfloor \xrightarrow{\overline{nk}\rhd m} n\lfloor k\lfloor\bullet\rfloor\rfloor \parallel m\lfloor q\rfloor}{nk \rhd \overline{m}.p \parallel n\lfloor k\lfloor q\rfloor\rfloor \parallel m\lfloor\bullet\rfloor \xrightarrow{\tau} p \parallel n\lfloor m\lfloor\bullet\rfloor\rfloor \parallel n'\lfloor q\rfloor}$$

Labelled Transition Semantics – Mobility (II)

(co-enter)

$$\frac{p_1 \xrightarrow{\delta_1 \rhd \langle q \rangle} p_1' \quad p_2 \xrightarrow{\delta_1 \rhd \overline{\delta_2}} p_2'}{p_1 \parallel p_2 \xrightarrow{\langle q \rangle \rhd \overline{\delta_2}} p_1' \parallel p_2'}$$

(co-exit)

$$\frac{p_2 \xrightarrow{\delta_1 \rhd \overline{\delta_2}} p_2' \quad p_1 \xrightarrow{(q) \rhd \delta_2} p_1'}{p_1 \parallel p_2 \xrightarrow{\delta_1 \rhd (q)} p_1' \parallel p_2'}$$

Labelled Transition Semantics – Mobility (II)

(co-enter)
$$\frac{p_1 \xrightarrow{\delta_1 \rhd \langle q \rangle} p_1' \quad p_2 \xrightarrow{\delta_1 \rhd \overline{\delta_2}} p_2'}{p_1 \parallel p_2 \xrightarrow{\langle q \rangle \rhd \overline{\delta_2}} p_1' \parallel p_2'}$$

(co-exit)
$$\frac{p_2 \xrightarrow{\delta_1 \rhd \overline{\delta_2}} p_2' \quad p_1 \xrightarrow{(q) \rhd \delta_2} p_1'}{p_1 \parallel p_2 \xrightarrow{\delta_1 \rhd (q)} p_1' \parallel p_2'}$$

$$\frac{\dfrac{}{n \rhd \overline{m}.p \xrightarrow{n \rhd \overline{m}} p} \quad \dfrac{}{m\lfloor \bullet \rfloor \xrightarrow{(q) \rhd m} m\lfloor q \rfloor}}{n \rhd \overline{m}.p \parallel m\lfloor \bullet \rfloor \xrightarrow{n \rhd (q)} p \parallel m\lfloor q \rfloor} \; m \notin (fn(q) \cup \{n\})$$

$$(m)(n \rhd \overline{m}.p \parallel m\lfloor \bullet \rfloor) \xrightarrow{n \rhd (q)} (m)(p \parallel m\lfloor q \rfloor)$$

Labelled Transition Semantics – Scope extension

(open)
$$\frac{p \xrightarrow{(\tilde{n})\overline{\delta}\triangleright\langle q\rangle} p'}{(n)p \xrightarrow{(n\tilde{n})\overline{\delta}\triangleright\langle q\rangle} p'} \; n\in fn(q)\setminus(fn(\delta)\cup\tilde{n})$$

(sync)
$$\frac{p_1 \xrightarrow{(\tilde{n})\overline{\pi}} p_1' \quad p_2 \xrightarrow{\pi} p_2'}{p_1 \parallel p_2 \xrightarrow{\tau} (\tilde{n})(p_1' \parallel p_2')} \; fn(p_2)\cap\tilde{n}=\emptyset$$

Labelled Transition Semantics – Scope extension

$$\text{(open)}\ \frac{p \xrightarrow{(\tilde{n})\overline{\delta}\rhd\langle q\rangle} p'}{(n)p \xrightarrow{(n\tilde{n})\overline{\delta}\rhd\langle q\rangle} p'}\ n\in fn(q)\setminus(fn(\delta)\cup\tilde{n}) \qquad \text{(sync)}\ \frac{p_1 \xrightarrow{(\tilde{n})\overline{\pi}} p_1' \quad p_2 \xrightarrow{\pi} p_2'}{p_1 \parallel p_2 \xrightarrow{\tau} (\tilde{n})(p_1' \parallel p_2')}\ fn(p_2)\cap\tilde{n}=\emptyset$$

$$\frac{\dfrac{\dfrac{\cdots}{n\lfloor\overline{k}\rfloor \parallel nk \xrightarrow{\overline{n}\rhd\langle\overline{k}\rangle} n\lfloor\bullet\rfloor \parallel nk}}{(k)(n\lfloor\overline{k}\rfloor \parallel nk) \xrightarrow{(k)\overline{n}\rhd\langle\overline{k}\rangle} n\lfloor\bullet\rfloor \parallel nk} \qquad \frac{\cdots}{n \rhd \overline{m} \parallel m\lfloor\bullet\rfloor \xrightarrow{n\rhd(\overline{k})} m\lfloor\overline{k}\rfloor}}{(k)(n\lfloor\overline{k}\rfloor \parallel nk) \parallel n \rhd \overline{m} \parallel m\lfloor\bullet\rfloor \xrightarrow{\tau} (k)(n\lfloor\bullet\rfloor \parallel nk \parallel m\lfloor\overline{k}\rfloor)}$$

Resource receptors replace higher order *exit* and *co-enter* actions.

$$\frac{p \xrightarrow{(\tilde{n})\langle q\rangle \triangleright \overline{\delta}} p'}{p \xrightarrow{\triangleright \overline{\delta}(\mathscr{D}_\delta)} (\tilde{n})\big(\, p' \parallel \mathscr{D}_\delta(q)\,\big)}\; fn(\mathscr{D}_\delta)\cap\tilde{n}=\emptyset$$

$$\frac{p \xrightarrow{(\tilde{n})\overline{\delta'} \triangleright \langle q\rangle} p'}{p \xrightarrow{(\mathscr{C}_\gamma)\overline{\delta'}\triangleright(\mathscr{D}_\delta)} (\tilde{n})(\,\mathscr{C}_\gamma(p') \parallel \mathscr{D}_\delta(q)\,)}\; (fn(\mathscr{C}_\gamma)\cup fn(\mathscr{D}_\delta))\cap\tilde{n}=\emptyset$$

LTS: The rules

(exit)

$$n\lfloor p \rfloor_m \xrightarrow{\overline{n}\rhd\langle p\rangle d} n\lfloor \bullet \rfloor_m$$

(enter)

$$n\lfloor \bullet \rfloor_m \xrightarrow{(p)\rhd n} n\lfloor p \rfloor_m$$

(give)

$$\frac{p_1 \xrightarrow{(\tilde{n})\overline{\delta_1}\rhd\langle q\rangle} p_1' \quad p_2 \xrightarrow{\delta_1\rhd\overline{\delta_2}} p_2' \qquad fn(p_2)\cap\tilde{n}=\emptyset}{p_1 \parallel p_2 \xrightarrow{(\tilde{n})\langle q\rangle\rhd\overline{\delta_2}} p_1' \parallel p_2'}$$

(take)

$$\frac{p_2 \xrightarrow{\delta_1\rhd\overline{\delta_2}} p_2' \quad p_1 \xrightarrow{(q)\rhd\delta_2} p_1'}{p_1 \parallel p_2 \xrightarrow{\delta_1\rhd(q)} p_1' \parallel p_2'}$$

(co-move)

$$\frac{p_1 \xrightarrow{(\tilde{n})\overline{\delta_1}\rhd\langle q\rangle} p_1' \quad p_2 \xrightarrow{(q)\rhd\delta_2} p_2' \qquad fn(p_2)\cap\tilde{n}=\emptyset}{p_1 \parallel p_2 \xrightarrow{\overline{\delta_1}\rhd\delta_2} (\tilde{n})(p_1' \parallel p_2')}$$

(nesting)

$$\frac{p \xrightarrow{\pi} p'}{n\lfloor p \rfloor_m \xrightarrow{n\cdot(\pi)} n\lfloor p' \rfloor_m}$$

(delete)

$$\frac{}{\tilde{n}\lfloor r \rfloor_m \xrightarrow{\overline{\natural m}} \mathbf{0}}$$

(alpha)

$$\frac{p \equiv_\alpha p' \quad p' \xrightarrow{\pi} q}{p \xrightarrow{\pi} q}$$

(open1)

$$\frac{p \xrightarrow{(\tilde{n})\overline{\delta}\rhd\langle q\rangle} p' \qquad n\in fn(q)\setminus(fn(\delta)\cup\tilde{n})}{(n)p \xrightarrow{(n\tilde{n})\overline{\delta}\rhd\langle q\rangle} p'}$$

(open2)

$$\frac{p \xrightarrow{(\tilde{n})\langle q\rangle\rhd\overline{\delta}} p' \qquad n\in fn(q)\setminus(fn(\delta)\cup\tilde{n})}{(n)p \xrightarrow{(n\tilde{n})\langle q\rangle\rhd\overline{\delta}} p'}$$

MR: Labelled Transition Bisimulation

- Bisimulation $\sim$ is the largest symmetric relation such that if $p \sim q$ then

$$p \xrightarrow{\psi} p' \text{ implies } \exists q \xrightarrow{\psi} q' \text{ such that } p' \sim q'$$

Thm: $\sim$ is a congruence.

Thm: $\sim_b = \sim$.

MR: Labelled Transition Bisimulation

- **Bisimulation** $\sim$ is the largest symmetric relation such that if $p \sim q$ then

$$p \xrightarrow{\psi} p' \text{ implies } \exists q \xrightarrow{\psi} q' \text{ such that } p' \sim q'$$

Thm: $\sim$ is a congruence.

Thm: $\sim_b = \sim$.

Note: To prove that $p \sim q$ we do not need to prove it for all contexts (congruence).

- **Exercise:** Resources cannot be copied (revisited):

$$(b)\big(\, a\lfloor b \rfloor \;\|\; !a\bar{b}.c \,\big) \sim (b)\big(\, a\lfloor b \rfloor \;\|\; a\bar{b}.c \,\big)$$

A Copy Capability

- Let us consider a copy capability that allows duplication of resources.

$$n\lfloor r \rfloor \parallel n \rhd\!\gg \overline{m}.p \parallel m\lfloor \bullet \rfloor \searrow n\lfloor r \rfloor \parallel p \parallel m\lfloor r \rfloor$$

- This is interesting in order to provide a finer control of copying: We allow slots to declare if they allow to be copied, and to what slots, using a Type System.

A Copy Capability

- Let us consider a copy capability that allows duplication of resources.

$$n\lfloor r \rfloor \parallel n \rhd\!\gg \overline{m}.p \parallel m\lfloor \bullet \rfloor \searrow n\lfloor r \rfloor \parallel p \parallel m\lfloor r \rfloor$$

- This is interesting in order to provide a finer control of copying: We allow slots to declare if they allow to be copied, and to what slots, using a Type System.

What is a Type System?

- Type systems are formal systems that assign types to terms. Types classify terms. Most famous application: 'terms are good if integers are never confused with booleans'. Since the classification should be stable under reductions, if p is good and $p \searrow^* q$, we expect q to be good. This fundamental property is called Subject reduction. Type systems are often specified by inference rules, which allow to 'deduce' type assignments.

A Copy Capability

- Let us consider a copy capability that allows duplication of resources.

$$n\lfloor r \rfloor \parallel n \rhd\!\gg \overline{m}.p \parallel m\lfloor \bullet \rfloor \searrow n\lfloor r \rfloor \parallel p \parallel m\lfloor r \rfloor$$

- This is interesting in order to provide a finer control of copying: We allow slots to declare if they allow to be copied, and to what slots, using a Type System.

What is a Type System?

- Type systems are formal systems that assign types to terms. Types classify terms. Most famous application: 'terms are good if integers are never confused with booleans'. Since the classification should be stable under reductions, if p is good and $p \searrow^* q$, we expect q to be good. This fundamental property is called Subject reduction. Type systems are often specified by inference rules, which allow to 'deduce' type assignments.

- In our case, we classify processes as good if they don't attempt to copy resources (and more) unless explicitly permitted.

MR: A Type System for Resource Control

$\pi ::= [\mathsf{S}, \mathsf{T}, \mathsf{M}, \mathsf{C}, \mathsf{X}, \mathsf{D}]$ process types

performs synchronisation with names in S

performs moving/copying to names in T

performs moving resources from names in M

performs copying from names in C

performs actions crossing names in X

performs deletion of names in D

$\nu ::= \pi_\natural$ name types

replace 'performs' with 'allows' and can be deleted according to $\natural$

$\natural ::= \Box \mid \triangle$ deletion flag

can (resp. cannot) be deleted

$\mu ::= [\mathsf{X}, a]$ path type

crosses names in X and acts on name a

- π is ordered componentwise and $\cup$ is defined accordingly.

MR: A Type System

- Environments:

$$\mathsf{E} ::= \emptyset \mid \mathsf{E}, n : \nu$$

- Judgments:

$\mathsf{E} \vdash \diamond$	good environment
$\mathsf{E} \vdash \nu$	good name type ν
$\mathsf{E} \vdash \mu$	good path type μ
$\mathsf{E} \vdash \pi$	good group and process type π
$\mathsf{E} \vdash \delta : \mu$	good path δ of type μ
$\mathsf{E} \vdash \lambda : \pi$	good label λ of type π
$\mathsf{E} \vdash p : \pi$	good process p of type π

MR: A Type System: Paths and Labels

(Path Start)

$$\frac{\mathsf{E}, a : \nu \vdash \diamond}{\mathsf{E}, a : \nu \vdash \alpha : [\varnothing, a]}\ \alpha \in \{a, \overline{a}\}$$

(Path Formation)

$$\frac{\mathsf{E}, n : \nu \vdash \delta : [\mathsf{X}, a]}{\mathsf{E}, n : \nu \vdash n\delta : [\mathsf{X} \cup \{n\}, a]}$$

(Label co-Sync)

$$\frac{\mathsf{E}, n : \nu \vdash \delta : [\mathsf{X}, a]}{\mathsf{E}, n : \nu \vdash \delta\overline{n} : [\{n\}, \varnothing, \varnothing, \varnothing, \mathsf{X}, \varnothing]}$$

(Label Sync)

$$\frac{\mathsf{E}, n : \nu \vdash \delta : [\mathsf{X}, a]}{\mathsf{E}, n : \nu \vdash \delta n : [\{n\}, \varnothing, \varnothing, \varnothing, \mathsf{X}, \varnothing]}$$

(Label Move)

$$\frac{\mathsf{E}, a : \pi_{\natural}, b : \pi'_{\natural'} \vdash \delta : [\mathsf{X}, a] \quad \mathsf{E}, a : \pi_{\natural}, b : \pi'_{\natural'} \vdash \delta' : [\mathsf{X}', b] \quad \pi \leq \pi'}{\mathsf{E}, a : \pi_{\natural}, b : \pi'_{\natural'} \vdash \delta \rhd \overline{\delta'} : [\varnothing, \{b\}, \{a\}, \varnothing, \mathsf{X} \cup \mathsf{X}'\varnothing]}$$

(Label Copy)

$$\frac{\mathsf{E}, a : \pi_{\natural}, b : \pi'_{\natural'} \vdash \delta : [\mathsf{X}, a] \quad \mathsf{E}, a : \pi_{\natural}, b : \pi'_{\natural'} \vdash \delta' : [\mathsf{X}', b] \quad \pi \leq \pi'}{\mathsf{E}, a : \pi_{\natural}, b : \pi'_{\natural'} \vdash \delta \rhd\!\gg \overline{\delta'} : [\varnothing, \{b\}, \varnothing, \{a\}, \mathsf{X} \cup \mathsf{X}'\varnothing]}$$

MR: A Type System: Processes

(Label Destroy)

$$\frac{\mathsf{E}, a : \pi_{\square} \vdash \delta : [\mathsf{X}, a]}{\mathsf{E} \vdash \natural\delta : [\varnothing, \varnothing, \varnothing, \varnothing, \mathsf{X}, \{a\}]}$$

(Proc $\mathbf{0}$)

$$\frac{\mathsf{E} \vdash \diamond}{\mathsf{E} \vdash \mathbf{0} : [\varnothing, \varnothing, \varnothing, \varnothing, \varnothing, \varnothing]}$$

(Res $\bullet$)

$$\frac{\mathsf{E} \vdash \diamond}{\mathsf{E} \vdash \bullet : [\varnothing, \varnothing, \varnothing, \varnothing, \varnothing, \varnothing]}$$

(Proc Pref)

$$\frac{\mathsf{E} \vdash \lambda : \pi \quad \mathsf{E} \vdash p : \pi'}{\mathsf{E} \vdash \lambda \,.\, p : \pi \biguplus \pi'}$$

(Proc Par)

$$\frac{\mathsf{E} \vdash p : \pi \quad \mathsf{E} \vdash q : \pi'}{\mathsf{E} \vdash p \parallel q : \pi \biguplus \pi'}$$

(Proc Rep)

$$\frac{\mathsf{E} \vdash p : \pi}{\mathsf{E} \vdash !p : \pi}$$

(Proc Res)

$$\frac{\mathsf{E}, n : \nu \vdash p : \pi}{\mathsf{E} \vdash (n : \nu)p : \pi \setminus n}$$

(Proc Slot)

$$\frac{\mathsf{E}, n : \pi_{\natural} \vdash r : \pi' \quad \pi' \leq \pi}{\mathsf{E} \vdash n\lfloor r \rfloor : [\varnothing, \varnothing, \varnothing, \varnothing, \varnothing, \varnothing]}$$

Subject Reduction

- **Thm.** If $\mathsf{E} \vdash p : \pi$ and $p \searrow q$, then $\mathsf{E} \vdash q : \pi'$ with $\pi' \leq \pi$
- **Exercise:** Can you devise an algorithm of type inference? That is, given a process p,

$$\mathscr{A}(p) = \langle \mathsf{E}, \pi \rangle \text{ such that } \mathsf{E} \vdash p : \pi \ldots$$

— Part II —

De Dimensionibus, Capacitatis et Mobilitate Calculus

———

A Criticism to MR

The model is fitting for the area it was meant to work for, but it doesn't scale up to other situations where you want resources to matter.

A Criticism to MR

The model is fitting for the area it was meant to work for, but it doesn't scale up to other situations where you want resources to matter.
Main criticisms:

- Not realistic on the space occupation. All processes take one slot.

$$n\lfloor \text{big\_and\_fat\_P} \rfloor \parallel m\lfloor \bullet \rfloor \parallel n\lfloor \text{small\_and\_slim\_P} \rfloor \parallel n \rhd \overline{m}$$

- Replication is not handled appropriately

$$a\lfloor !P \rfloor = a\lfloor !P \parallel P \rfloor = a\lfloor !P \parallel P \parallel P \rfloor = a\lfloor !P \parallel P \parallel P \parallel P \rfloor = \ldots$$

- It doesn't really allow an analisys of variation in space occupation:

Computation takes space, dynamically, and we'd like to model it.

A Calculus of Bounded Capacities: Movement

Fundamentals: Space Conscious Movement

$$a[\,\mathbf{in}\,b\,.\,P \mid Q\,] \mid b[\,\_ \mid R\,] \quad \searrow \quad \_ \mid b[\,a[\,P \mid Q\,] \mid R\,]$$

$$\_ \mid b[\,a[\,\mathbf{out}\,b\,.\,P \mid Q\,] \mid R\,] \quad \searrow \quad a[\,P \mid Q\,] \mid b[\,\_ \mid R\,]$$

A Calculus of Bounded Capacities: Movement

Fundamentals: Space Conscious Movement

$$a[\,\mathbf{in}\,b\,.\,P \mid Q\,] \mid b[\,\_ \mid R\,] \quad \searrow \quad \_ \mid b[\,a[\,P \mid Q\,] \mid R\,]$$

$$\_ \mid b[\,a[\,\mathbf{out}\,b\,.\,P \mid Q\,] \mid R\,] \quad \searrow \quad a[\,P \mid Q\,] \mid b[\,\_ \mid R\,]$$

Example: Travelling needs but consumes no space.

$$a[\,\mathbf{in}\,b\,.\,\mathbf{in}\,c\,.\,\mathbf{out}\,c\,.\,\mathbf{out}\,b\,.\,\mathbf{0}\,] \mid b[\,\_ \mid c[\,\_\,]\,]$$

$$\searrow\searrow \_ \mid b[\,\_ \mid c[\,a[\,\mathbf{out}\,c\,.\,\mathbf{out}\,b\,.\,\mathbf{0}\,]\,]\,]$$

$$\searrow\searrow a[\,\mathbf{0}\,] \mid b[\,\_ \mid c[\,\_\,]\,]$$

A Calculus of Bounded Capacities: Replication

Fundamentals: Space Conscious Replication

$$!^k P \mid \overbrace{\_ \mid \ldots \mid \_}^{k \text{ times}} \quad \equiv \quad !^k P \mid P$$

What is the $!^k$? A type annotation depending on the size of P. Gives a typed reduction: (1) types appear as conditions on reductions; (2) some of the calculus' operators make only sense with type annotations.

Notation. We use $\_^k$ as a shorthand for $\overbrace{\_ \mid \ldots \mid \_}^{k \text{ times}}$.

A Calculus of Bounded Capacities: Replication

Fundamentals: Space Conscious Replication

$$!^k P \mid \overbrace{\_ \mid \ldots \mid \_}^{k \text{ times}} \quad \equiv \quad !^k P \mid P$$

What is the $!^k$? A type annotation depending on the size of P. Gives a typed reduction: (1) types appear as conditions on reductions; (2) some of the calculus' operators make only sense with type annotations.

Notation. We use $\_^k$ as a shorthand for $\overbrace{\_ \mid \ldots \mid \_}^{k \text{ times}}$.

Example: Simple recursion: rec $(x)p$ in q

$$\ulcorner p \urcorner \triangleq \text{substitute } x \text{ with } \_^k \text{ in } p$$

Then $\langle\!\langle \text{rec } (x)p \text{ in } q \rangle\!\rangle \triangleq !^k \ulcorner p \urcorner \mid \ulcorner q \urcorner$ (assuming p has 'size' k and q 'nice', i.e. spawns no space-grabbing actions in parallel with x).

A Calculus of Bounded Capacities: Open

Fundamentals: Space Conscious Incremental Open

It would be possible to use

$$\blacksquare^k \mid \mathbf{opn}\, a \,.\, P \mid a^k[\, Q \,] \quad \searrow \quad Q \mid P \mid \blacksquare$$

but it would not be tight enough: in spite of static types, ambients may change size dynamically. In order to have better control on space usage, we use:

$$\blacksquare^k \mid \mathbf{opn}\, a \,.\, P \mid a[\, \uparrow^k Q \mid R \,] \quad \searrow \quad Q \mid P \mid a[\, R \,]$$

together with a garbage collection rule $n[\, \mathbf{0} \,] \equiv \blacksquare$.

Two readings:

- (1) step-wise refinement of open;
- (2) spawning of process in the father's ambient.

BoCa: Open (Examples)

Example: Locks:

$$\mathsf{lock}\ n\,.\,p \triangleq \mathbf{opn}\,n\,.\,p$$

$$\mathsf{unlock}\ n\,.\,p \triangleq n[\,\uparrow\mathbf{0}\,] \mid p$$

BoCa: Open (Examples)

Example: Locks:

$$\mathsf{lock}\ n\,.\,p \triangleq \mathbf{opn}\,n\,.\,p$$

$$\mathsf{unlock}\ n\,.\,p \triangleq n[\ \uparrow\mathbf{0}\ ]\mid p$$

Example: Communication Channels:

$\mathsf{from\_a\_to\_b}(m).p \triangleq \mathsf{ch}[\ \mathbf{out}\,a\,.\,\mathbf{in}\,b\,.\uparrow^k\langle M\rangle\ ]$ (b and the father ambient must provide suitable space for ch to travel; b must perform $\mathbf{opn}\,\mathsf{ch}$).

BoCa: Open (Examples)

Example: Locks:

$$\mathsf{lock}\ n\,.\,p \triangleq \mathbf{opn}\,n\,.\,p$$

$$\mathsf{unlock}\ n\,.\,p \triangleq n[\ \uparrow\mathbf{0}\ ] \mid p$$

Example: Communication Channels:

$\mathsf{from\_a\_to\_b}(m).p \triangleq \mathsf{ch}[\ \mathbf{out}\,a\,.\,\mathbf{in}\,b\,.\uparrow^k\langle M\rangle\ ]$ (b and the father ambient must provide suitable space for ch to travel; b must perform $\mathbf{opn}\,\mathsf{ch}$).

Example: Simple Recursion: $\mathsf{rec}\ (x_1)p_1 \ldots (x_n)p_n\ \mathsf{in}\ q$

$$\ulcorner p \urcorner \triangleq \text{substitute } x_i \text{ with } x_i[\ \uparrow 0\ ] \mid \_^{k_i} \text{ in } p$$

Then $\langle\!\langle \mathsf{rec}\ (x_1)p_1 \ldots (x_n)p_n\ \mathsf{in}\ q \rangle\!\rangle \triangleq (\ \Pi_i !^{k_i}\mathbf{opn}\,x_i\,.\ulcorner p_i\urcorner\ ) \mid \ulcorner q \urcorner$ (assuming p_i has 'size' k_i and q 'nice', i.e. no space-grabbing actions in parallel with x).

A Calculus of Bounded Capacities: Transfer

Fundamentals: Space Acquisition and Release

$$\blacksquare \mid \mathbf{tr}\, a\,.\, P \mid a[\, \overline{\mathbf{tr}}\hat{}\,.\, Q \mid R\,] \quad \searrow \quad P \mid a[\, Q \mid R \mid \blacksquare\,]$$

$$\overline{\mathbf{tr}}\, a\,.\, P \mid a[\, \mathbf{tr}\hat{}\,.\, Q \mid \blacksquare \mid R\,] \quad \searrow \quad \blacksquare \mid P \mid a[\, Q \mid R\,]$$

A Calculus of Bounded Capacities: Transfer

Fundamentals: Space Acquisition and Release

$$\_ \mid \mathbf{tr}\, a\,.\,P \mid a[\, \overline{\mathbf{tr}}\hat{}\,.\,Q \mid R\,] \quad \searrow \quad P \mid a[\, Q \mid R \mid \_\,]$$

$$\overline{\mathbf{tr}}\, a\,.\,P \mid a[\, \mathbf{tr}\hat{}\,.\,Q \mid \_ \mid R\,] \quad \searrow \quad \_ \mid P \mid a[\, Q \mid R\,]$$

Example: A Memory Module

$$\mathsf{memMod} \triangleq \_^{256MB} \mid \_ \mid !\overline{\mathbf{tr}}\,\mathsf{malloc}\,.\,\_ \mid !\mathbf{tr}\,\mathsf{free}\,.\,\_$$

A Calculus of Bounded Capabilities: Syntax

$$P ::= \_ \mid \mathbf{0} \mid M\,.\,P \mid P \mid P \mid M[\,P\,] \mid !^k P \mid \uparrow^k P \mid (\boldsymbol{\nu} n : \pi)P \mid (x : \chi)P \mid \langle M \rangle P$$

$$M ::= \varepsilon \mid x \mid a \mid M\,.\,M \mid \mathbf{in}\,M \mid \mathbf{out}\,M \mid \mathbf{opn}\,M \mid \mathbf{tr}\,\eta \mid \overline{\mathbf{tr}}\,\eta$$

$$\eta ::= a \mid \hat{}$$

A Calculus of Bounded Capabilities: Syntax

$$P ::= \blacksquare \mid \mathbf{0} \mid M\,.\,P \mid P \mid P \mid M[\,P\,] \mid !^k P \mid \uparrow^k P \mid (\boldsymbol{\nu} n : \pi)P \mid (x : \chi)P \mid \langle M \rangle P$$

$$M ::= \varepsilon \mid x \mid a \mid M\,.\,M \mid \mathbf{in}\,M \mid \mathbf{out}\,M \mid \mathbf{opn}\,M \mid \mathbf{tr}\,\eta \mid \overline{\mathbf{tr}}\,\eta$$

$$\eta ::= a \mid \hat{\ }$$

Structural Congruence:

$(|, \mathbf{0})$ is a commutative monoid.

$$(\boldsymbol{\nu} a)(P \mid Q) \equiv (\boldsymbol{\nu} a)P \mid Q \qquad \text{if } a \notin fn(Q)$$

$$(\boldsymbol{\nu} a)\mathbf{0} \equiv \mathbf{0}$$

$$(\boldsymbol{\nu} a)\langle M \rangle P \equiv \langle M \rangle (\boldsymbol{\nu} a)P \qquad \text{if } a \notin fn(P)$$

$$(\boldsymbol{\nu} a)(\boldsymbol{\nu} b)P \equiv (\boldsymbol{\nu} b)(\boldsymbol{\nu} a)P$$

$$a[\,(\boldsymbol{\nu} b)P\,] \equiv (\boldsymbol{\nu} b)a[\,P\,] \qquad \text{if } a \neq b$$

$$!^0 \mathbf{0} \equiv \mathbf{0}$$

$$!^k P \mid \blacksquare^k \equiv !^k P \mid P$$

$$n[\,\mathbf{0}\,] \equiv \blacksquare$$

BoCa: Reduction Semantics

$$(R\text{-}\mathbf{in}) \quad a[\,\mathbf{in}\,b\,.\,P \mid Q\,] \mid b[\,\_ \mid R\,] \searrow \_ \mid b[\,a[\,P \mid Q\,] \mid R\,]$$

$$(R\text{-}\mathbf{out}) \quad \_ \mid b[\,a[\,\mathbf{out}\,b\,.\,P \mid Q\,] \mid R\,] \searrow a[\,P \mid Q\,] \mid b[\,\_ \mid R\,]$$

$$(R\text{-}\mathbf{opn}) \quad \_^k \mid \mathbf{opn}\,a\,.\,P \mid a[\,\uparrow^k Q \mid R\,] \searrow Q \mid P \mid a[\,R\,]$$

$$(R\text{-}\mathbf{tr}_{\,down}) \quad \_ \mid \mathbf{tr}\,a\,.\,P \mid a[\,\overline{\mathbf{tr}}\hat{}\,.\,Q \mid R\,] \searrow P \mid a[\,Q \mid R \mid \_\,]$$

$$(R\text{-}\mathbf{tr}^{\,up}) \quad \overline{\mathbf{tr}}\,a\,.\,P \mid a[\,\mathbf{tr}\hat{}\,.\,Q \mid \_ \mid R\,] \searrow \_ \mid P \mid a[\,Q \mid R\,]$$

$$(R\text{-}comm) \quad (x:\chi)P \mid \langle M\rangle Q \searrow P\{x \leftarrow M\} \mid Q$$

A System of Capacity Types

Capacity Types: $\sigma, \phi, \ldots$ have pairs of ints $\langle m, M \rangle$, with $m \leq M$ and $0 \leq M$.
Notation: formal sum: $\sigma = k_1 \cdot \mathsf{m} + k_2 \cdot \mathsf{M}$, and $\sigma_{\mathsf{m}} = k_1$ and $\sigma_{\mathsf{M}} = k_1$.

Exchange Types: $\chi ::= Shh \mid \langle \sigma, \chi \rangle$

Process and **Ambient Types** are pairs of capacity and exchange types.

$\mathtt{amb} : \langle \sigma, \chi \rangle$	amb has no less than σ_{m} and no more than σ_{M} spaces
$\mathtt{proc} : \langle \sigma, \chi \rangle$	proc takes no less than σ_{m} and no more than σ_{M} spaces
$\mathtt{cap} : \phi$	cap transforms processes adding ϕ to their σ

Capacity types are ordered as follows:

$$\sigma \lessdot \tau \equiv \tau_{\mathsf{m}} \leq \sigma_{\mathsf{m}} \sqcup 0 \text{ and } \sigma_{\mathsf{M}} \leq \tau_{\mathsf{M}},$$

A Typing System: Capabilities

(Axiom)

$$\frac{}{\Gamma, n : \chi \vdash n : \chi}$$

(Slot)

$$\frac{}{\Gamma \vdash \blacksquare : \langle\langle 1, 1\rangle, \chi\rangle}$$

(In)

$$\frac{}{\Gamma \vdash \mathbf{in}\, M : \langle\langle 0, 0\rangle, \chi\rangle}$$

(Transf)

$$\frac{}{\Gamma \vdash \mathbf{tr}\, \eta : \langle -\mathsf{m}, \chi\rangle}$$

(Open)

$$\frac{}{\Gamma, M : \langle\sigma, \chi\rangle \vdash \mathbf{opn}\, M : \langle\langle 0, 0\rangle, \chi\rangle}$$

(Empty)

$$\frac{}{\Gamma \vdash \varepsilon : \langle\langle 0, 0\rangle, \chi\rangle}$$

(Zero)

$$\frac{}{\Gamma \vdash \mathbf{0} : \langle\langle 0, 0\rangle, \chi\rangle}$$

(Out)

$$\frac{}{\Gamma \vdash \mathbf{out}\, M : \langle\langle 0, 0\rangle, \chi\rangle}$$

(CoTransf)

$$\frac{}{\Gamma \vdash \overline{\mathbf{tr}}\, \eta : \langle +\mathsf{M}, \chi\rangle}$$

(Composition)

$$\frac{\Gamma \vdash M : \langle\phi, \chi\rangle \quad \Gamma \vdash M' : \langle\phi', \chi\rangle}{\Gamma \vdash M.M' : \langle\phi + \phi', \chi\rangle}$$

A Typing System: Processes

(Prefix)
$$\frac{\Gamma \vdash M : \langle \phi, \chi \rangle \quad \Gamma \vdash P : \langle \sigma, \chi \rangle}{\Gamma \vdash M \,.\, P : \langle \phi + \sigma, \chi \rangle}$$

(Parallel)
$$\frac{\Gamma \vdash P : \langle \sigma, \chi \rangle \quad \Gamma \vdash Q : \langle \sigma', \chi \rangle}{\Gamma \vdash P \mid Q : \langle \sigma + \sigma', \chi \rangle}$$

(Input)
$$\frac{\Gamma, x : \chi \vdash P : \langle \sigma, \chi \rangle}{\Gamma \vdash (x : \chi) P : \langle \sigma, \chi \rangle}$$

(Output)
$$\frac{\Gamma \vdash M : \chi \quad \Gamma \vdash P : \langle \sigma, \chi \rangle}{\Gamma \vdash \langle M \rangle P : \langle \sigma, \chi \rangle}$$

(New)
$$\frac{\Gamma, a : \pi \vdash P : \pi'}{\Gamma \vdash (\boldsymbol{\nu} a : \pi) P : \pi'}$$

(Ambient)
$$\frac{\Gamma, M : \langle \sigma', \chi \rangle \vdash P : \langle \sigma, \chi \rangle \quad \sigma \lessdot \sigma'}{\Gamma, M : \langle \sigma', \chi \rangle \vdash M[\ P\ ] : \langle \langle 1, 1 \rangle, \chi' \rangle}$$

(Spawn)
$$\frac{\Gamma \vdash P : \langle \langle n, n \rangle, \chi \rangle}{\Gamma \vdash \uparrow^n P : \langle \langle 0, 0 \rangle, \chi \rangle}$$

(Replication)
$$\frac{\Gamma \vdash P : \langle \langle n, n \rangle, \chi \rangle}{\Gamma \vdash !^n P : \langle \langle 0, 0 \rangle, \chi \rangle}$$

A Calculus of Bounded Capabilities

Thm: Subject Reduction

If $\Gamma \vdash P : \langle \sigma, \chi \rangle$ and $P \searrow Q$ then $\Gamma \vdash Q : \langle \sigma', \chi \rangle$ for some $\sigma' \leq \sigma$.

A Calculus of Bounded Capabilities

Thm: Subject Reduction

If $\Gamma \vdash P : \langle \sigma, \chi \rangle$ and $P \searrow Q$ then $\Gamma \vdash Q : \langle \sigma', \chi \rangle$ for some $\sigma' \leq \sigma$.

Exercise:

Prove that

$$\Gamma \vdash P : \langle \langle n, n \rangle, \chi \rangle \implies n \geq 0$$

Exercise:

What would happen if we considered well-typed the replication of processes like $\overline{\mathbf{tr}}\ \mathbf{.0}$ or $\mathbf{tr}\ \mathbf{.0}$? Consider for istance that the process $\mathbf{!tr}\ \mathbf{.0}$ would grant away all the free slots of the ambient.

Exercise: Consider each of the following processes and say whether they are typeable or dangerous/unsafe is some way.

$$\text{starvy} \triangleq n[\ \mathbf{!tr}\ \mathbf{.0}\ ]$$

$$\text{greedy} \triangleq n[\ \mathbf{!tr}\ \mathbf{.0} \mid \_\ ]$$

$$\text{floody} \triangleq n[\ \mathbf{!tr}\ \mathbf{.}\_ \mid \_\ ]$$

Some References

- J.Chr. Godskesen, T. Hildebrandt, V. Sassone, A Calculus of Mobile Resources, CONCUR 2002 and Tech Rep ITU Copenaghen.
- F. Barbanera, M. Bugliesi, M. Dezani, V. Sassone, A Calculus of Bounded Capacities (BoCa: De Dimensionibus, Capacitatis et Mobilitate), Work in progress.

- K. Crary, S. Weirich, Resource Bounds Certification, POPL 2000.
- W. Charatonik, A.D. Gordon, J.-M. Talbot, Finite Control Mobile Ambients, ESOP 2002.
- D. Teller, P. Zimmer, D. Hirschkoff, Using Ambients to Control Resources, CONCUR 2002 + Tech Rep ENS Lyon.

Conclusions

- In the large: Resource bounds negotiation and enforcement in GC.
- In the small: Expressivenss of MR and BoCa; Smarter types; ...
- In general: A lot to be done...

Conclusions

- In the large: Resource bounds negotiation and enforcement in GC.
- In the small: Expressivenss of MR and BoCa; Smarter types; ...
- In general: A lot to be done...

Acknowledgments. This lecture reports joint work with several collegues: Franco Barbanera, Michele Bugliesi, Mariangiola Dezani, Jens Chr. Godskesen, Thomas Hildebrandt.

Advert:

- 'R U a PhD Student?
- 'R U interested in the "Foundations of Global Computing"?
- 'R U willing to spend x months $(6 \leq x \leq 12)$ in Sussex

Would you like to be a Marie Curie Research Fellow?

Apply for a grant to the Marie Curie Traning Site "DisCo: Distributed Computation"

http://www.sussex.ac.uk/Units/pgrad/marie-curie/

http://www.susx.ac.uk/projects/disco