# Global Computing

WAN programming for building *global systems*.
They are hard to be made robust because:

- Absence of centralised control
  Client-Server not enough: P2P
- Administrative domains (Security)
- Interoperability
  - different platforms
  - different devices
    (e.g. PDA, laptop, mobile phones...)
- "Mobility" (resources and computation)
- ...

# Global Computing

WAN programming for building *global systems*.
They are hard to be made roboust because:

- *Network Awareness*
  - Applications are location dependent
  - Locations have different features
  - and allow multiple (security) policies
- Service Level Agreement
- Independently programmed in a distributed environment
- Reasoning on space and time
- ...

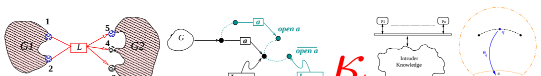
# Web Services: A programming metaphor

- Applications access *services* that must be
  - Published
  - Searched
  - Binded
- Services are
  - “Autonomous”
  - Independent (local choices, independently built)
  - Mobile/stationary
  - “Interconnected”
- Security issues: hostile environment

# WAN Foundations

$\pi$-calculus [?] (very basic wrt WAN)

- Ambient [?, ?, ?]
- Djoin [?, ?]
- D$\pi$ [?, ?]
- Klaim [?, ?, ?]
- Seal [?]
- ...



# A Model for Declarative WAN Programming

**In collaboration with G. Ferrari and U. Montanari**

# Hypergraphs Programming model[2]

- Graphs for distributed systems [?]
- Edge replacement for graph rewritings [?]
- Edge replacement/distributed constraint solving problem [?]
- Graphs grammars for software architecture styles [?]
- Synchronised Hyperedge Replacement (SHR) with mobility for name passing calculi [?]

# Hypergraphs Programming model[3]

We aim at tackling new *non-functional* computational phenomena of systems using SHR.
The metaphor is

- “WAN systems *as* Hypergraphs”
- “WAN computations *as* SHR”

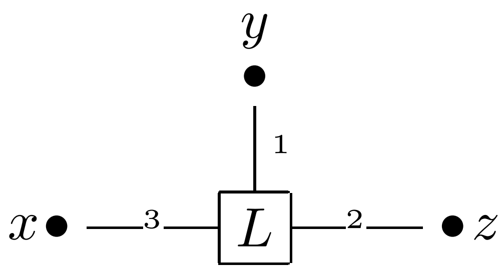
In other words:

- Components are represented by hyperedges
- Systems are *bunches* of (connected) hyperedges
- Computing means to rewrite hyperedge...
- ...according to a synchronisation policy

# Hyperedges and Hypergraphs Syntax

A hyperedge generalises edges: It connects more than two nodes

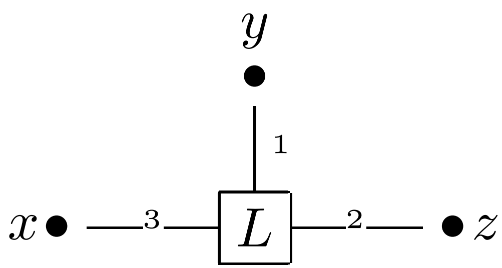
$$L : 3, \quad L(y, z, x),$$



# Hyperedges and Hypergraphs Syntax

A hyperedge generalises edges: It connects more than two nodes

$$L : 3, \quad L(y, z, x),$$

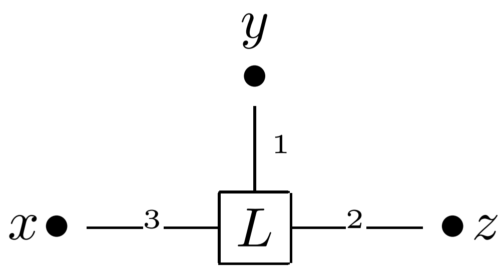



$$\begin{array}{rcl} G & ::= & nil \mid \nu\, y.G \\ & \mid & L(\vec{x}) \mid G|G \end{array}$$

# Hyperedges and Hypergraphs Syntax

A hyperedge generalises edges: It connects more than two nodes

$$L : 3, \quad L(y, z, x),$$



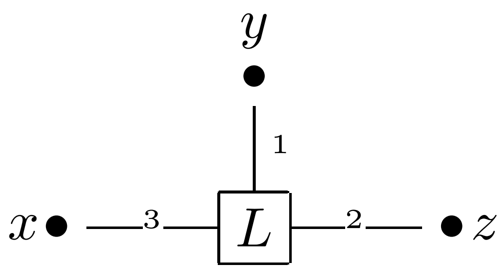

$$G \quad ::= \quad nil \mid \nu\, y.G \quad \mid \quad L(\vec{x}) \mid G|G$$

Syntactic Judgement $\quad \Gamma \vdash G, \qquad fn(G) \subseteq \Gamma$

# Hyperedges and Hypergraphs Syntax

A hyperedge generalises edges: It connects more than two nodes

$L : 3, \quad L(y, z, x),$




$$G \; ::= \; nil \;|\; \nu\, y.G \;|\; L(\vec{x}) \;|\; G|G$$

Syntactic Judgement $\quad \Gamma \vdash G, \quad fn(G) \subseteq \Gamma$

An example:

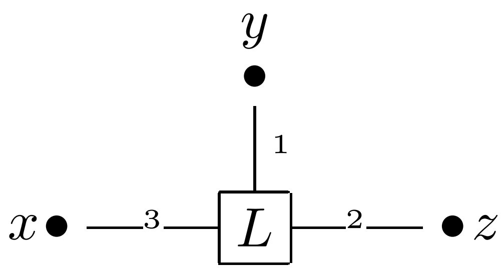
$$L : 3, \quad M : 2$$

$$x, y \vdash \nu\, z.(L(y, z, x)|M(y, z))$$

# Hyperedges and Hypergraphs Syntax

A hyperedge generalises edges: It connects more than two nodes

$L : 3, \quad L(y, z, x),$



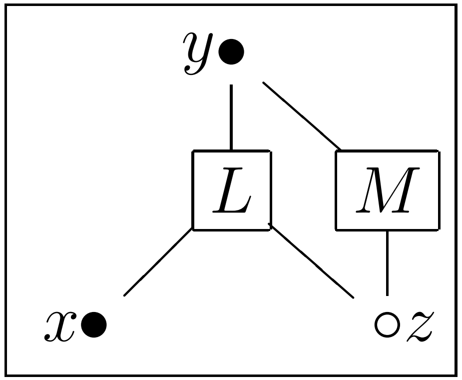

$$G \;::=\; nil \;\mid\; \nu\, y.G \;\mid\; L(\vec{x}) \;\mid\; G|G$$

**Syntactic Judgement** $\quad \Gamma \vdash G, \qquad fn(G) \subseteq \Gamma$

An example:

$L : 3, \quad M : 2$

$x, y \vdash \nu\, z.(L(y, z, x) | M(y, z))$

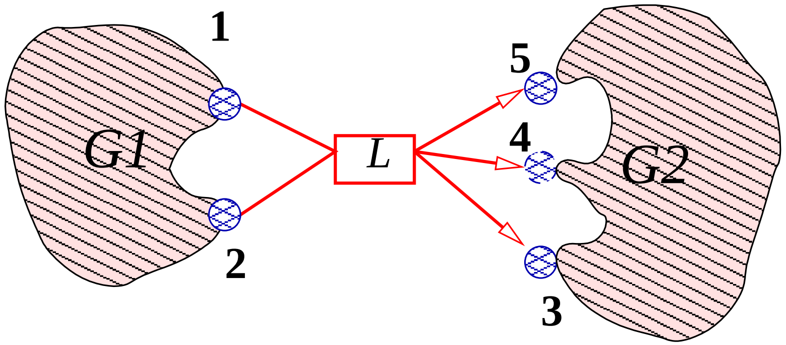


# Replacement of Hyperedges

$L \rightarrow G$

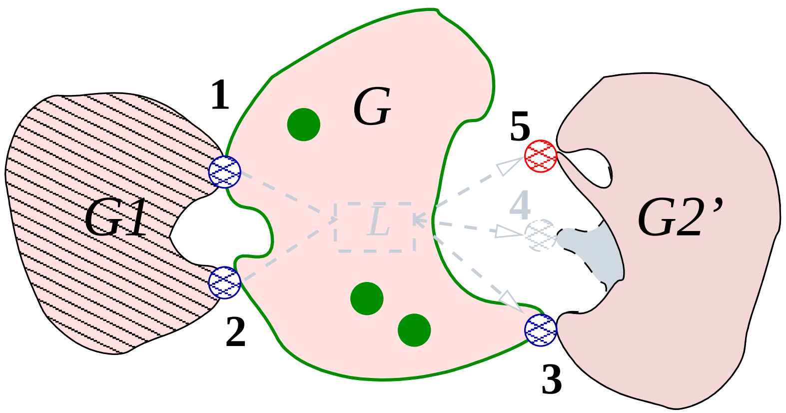
# Replacement of Hyperedges

$$L \longrightarrow G$$



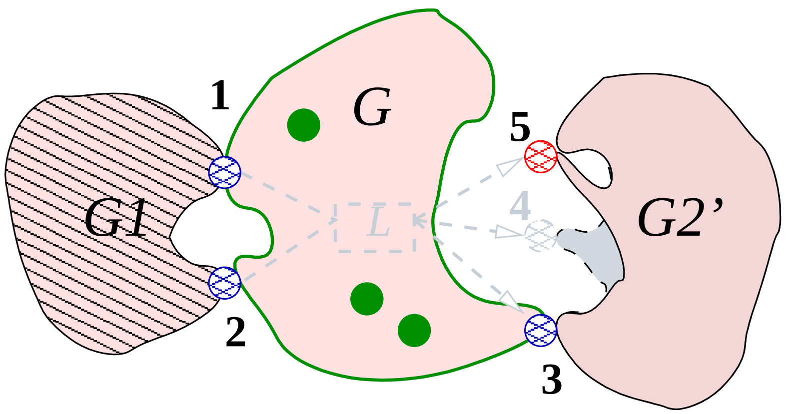
# Replacement of Hyperedges

$L \rightarrow G$



# Replacement of Hyperedges

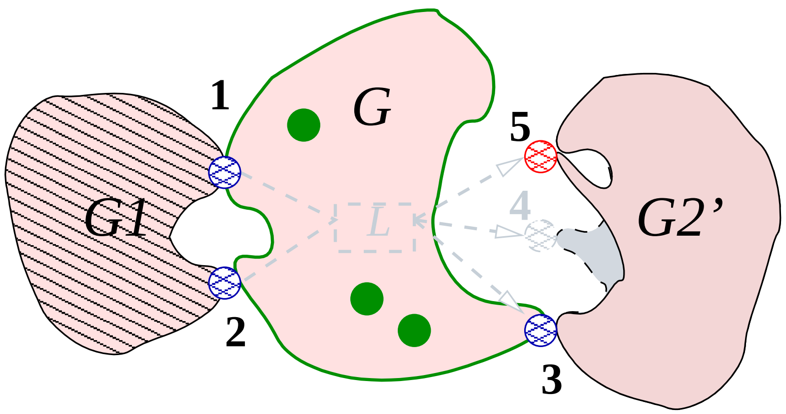
$L \to G$




- Edge replacement: local
- Synchronisation as distributed constraint solving
- Multiple synchronisation
- New node creation
- Node fusion: mobility model

# Replacement of Hyperedges

$L \to G$



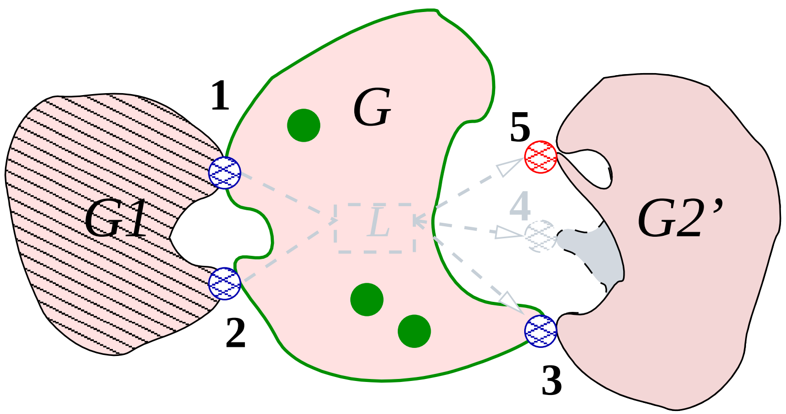

- Edge replacement: local
- Synchronisation as distributed constraint solving
- Multiple synchronisation
- New node creation
- Node fusion: mobility model

Benefits:

- Uniform framework for $\pi$, $\pi$-I, fusion
- LTS for Ambient ...
- ... for Klaim ...

# Replacement of Hyperedges

$L \rightarrow G$




- Edge replacement: local
- Synchronisation as distributed constraint solving
- Multiple synchronisation
- New node creation
- Node fusion: mobility model

Benefits:

- Uniform framework for $\pi$, $\pi$-I, fusion
- LTS for Ambient ...
- ... for Klaim ...
- ... and *path reservation* for Qlaim
- wireless networks
- expressive for distributed coordination

# Hypergraph Semantics: Productions

$$\underbrace{x_1, \ldots, x_n}_{X} \vdash L(x_1, \ldots, x_n) \xrightarrow[\pi]{\Lambda} \Gamma \vdash G,$$

- $\Lambda \subseteq X \times Act \times \mathcal{N}^*$ set of constraints
- $\pi : X \to X$ fusion substitution, i.e.

$$\forall x_i, x_j \in X.\pi(x_i) = x_j \Rightarrow \pi(x_j) = x_j$$

- $\Gamma = \pi(X) \cup (\mathrm{n}(\Lambda) \setminus X)$
- $\mathrm{fn}(G) \subseteq \Gamma$

# Hypergraph Semantics: Productions

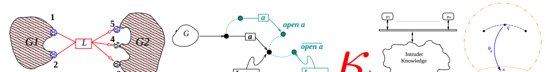
$$\underbrace{x_1, \ldots, x_n}_{X} \vdash L(x_1, \ldots, x_n) \xrightarrow[\pi]{\Lambda} \Gamma \vdash G,$$

- $\Lambda \subseteq X \times Act \times \mathcal{N}^*$ set of constraints
- $\pi : X \to X$ fusion substitution, i.e.

$$\forall x_i, x_j \in X. \pi(x_i) = x_j \Rightarrow \pi(x_j) = x_j$$

- $\Gamma = \pi(X) \cup (\mathrm{n}(\Lambda) \setminus X)$
- $\mathrm{fn}(G) \subseteq \Gamma$

Graph Rewritings: $\Gamma_1 \vdash G_1 \xrightarrow[\pi]{\Lambda} \Gamma_2 \vdash G_2$



# Applying the Model

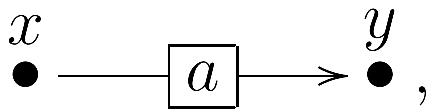
Ambient $a[\ldots] \,|\, open\ a \ \rightarrow \ \ldots$

# Applying the Model

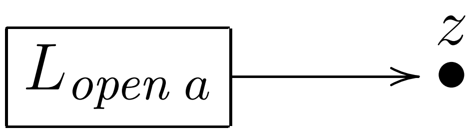
Ambient $\quad a[...] | open\ a \ \rightarrow \ ...$

Components

$a[\cdots]:$




$open\ a:$

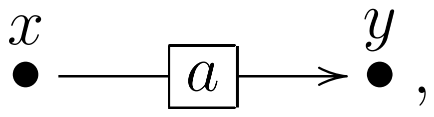


# Applying the Model

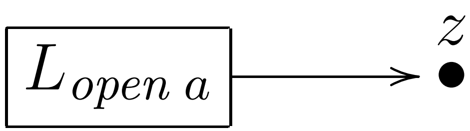
**Ambient** $a[...]|open\ a \;\to\; ...$

**Components**

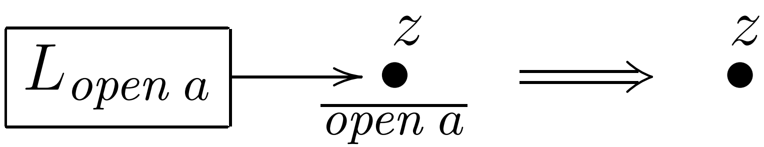
$a[\cdots]:$



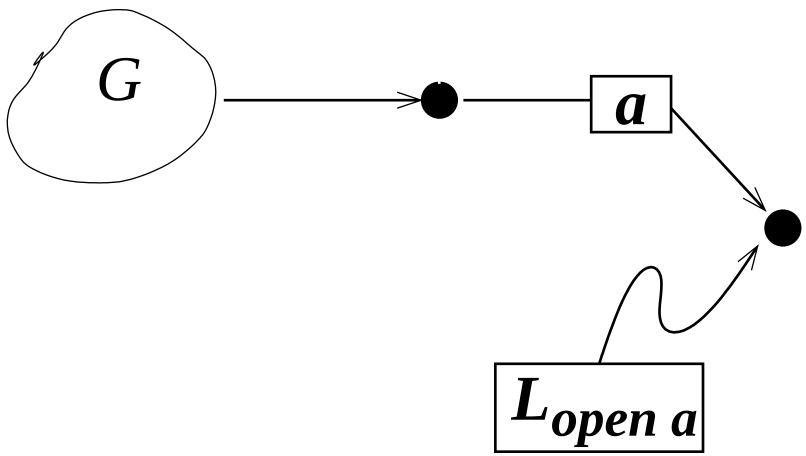

$open\ a:$



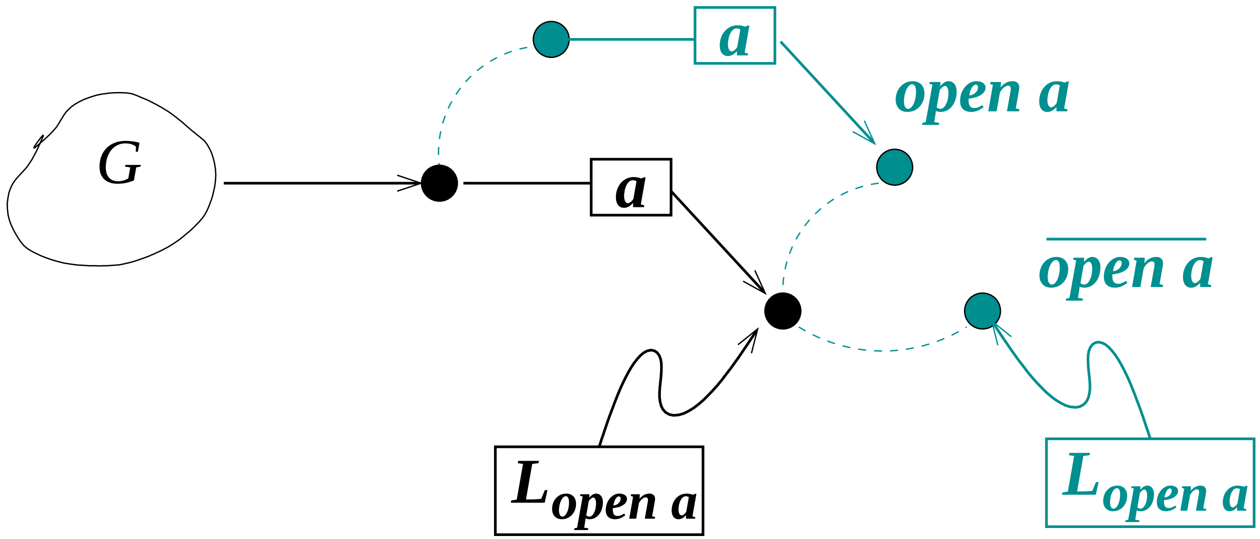

**Productions**

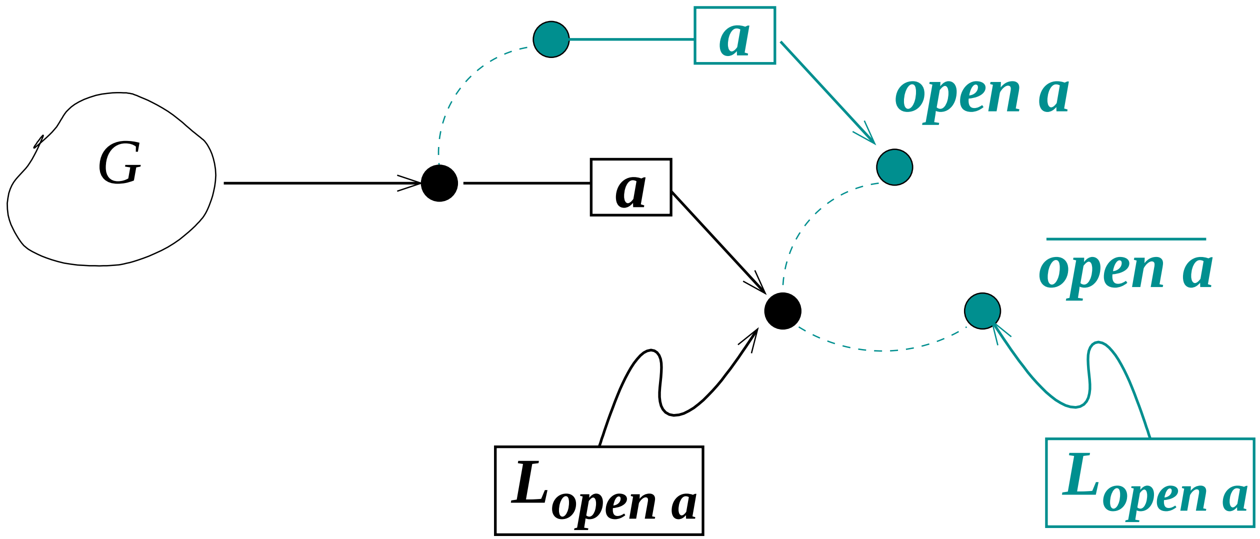
# Applying the Model: Node Fusion

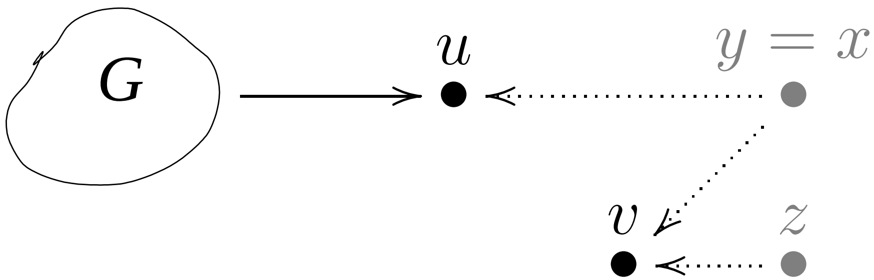


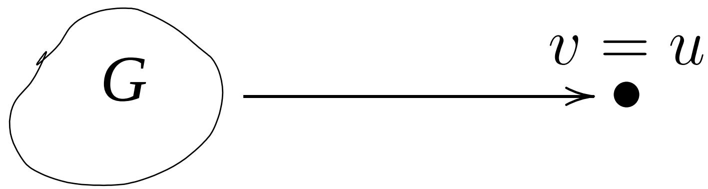
# Applying the Model: Node Fusion



# Applying the Model: Node Fusion



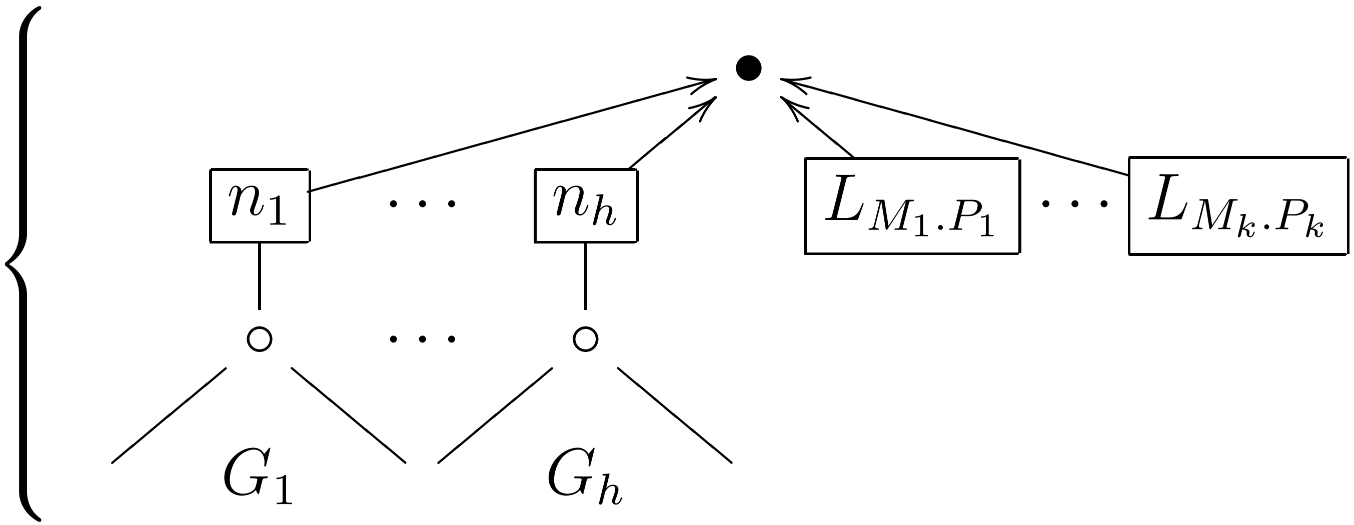



# Graphs and Ambient

$$
\begin{aligned}
[\![\, \mathbf{nil} \,]\!]_x &= x \vdash nil \\
[\![\, n[P] \,]\!]_x &= x \vdash \nu\, y.(G \mid n(y,x)), \quad \text{if } \; y \neq x \wedge [\![\, P \,]\!]_y = y \vdash G \\
[\![\, M.P \,]\!]_x &= x \vdash L_{M.P}(x) \\
[\![\, P_1 | P_2 \,]\!]_x &= x \vdash G_1 \mid G_2, \qquad \text{if } \; [\![\, P_i \,]\!]_x = x \vdash G_i \;\wedge\; i = 1,2 \\
[\![\, rec\, X.\, P \,]\!]_x &= [\![\, P[^{rec\, X.\, P}/_X] \,]\!]_x
\end{aligned}
$$

Ambient Graphs

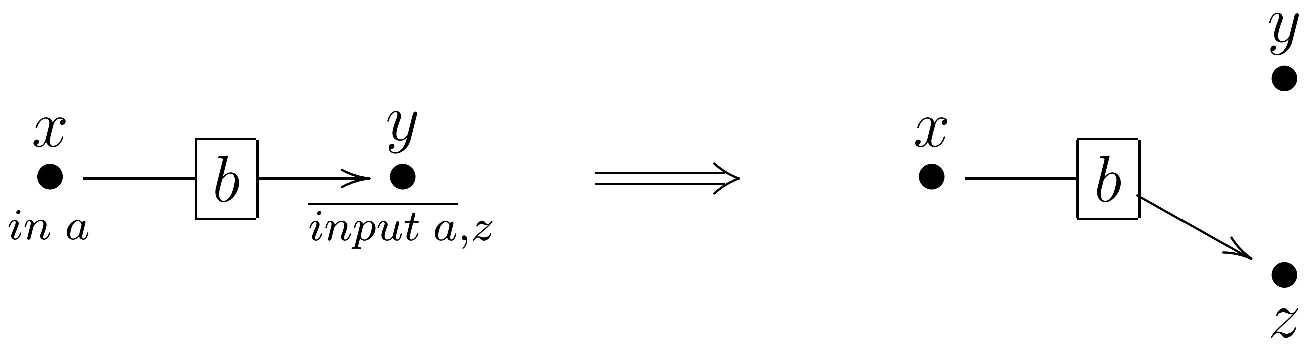



**Theorem** $[\![\, \_ \,]\!]_\_$ is a bijection on ambient graphs

# Coordination Productions for Ambient

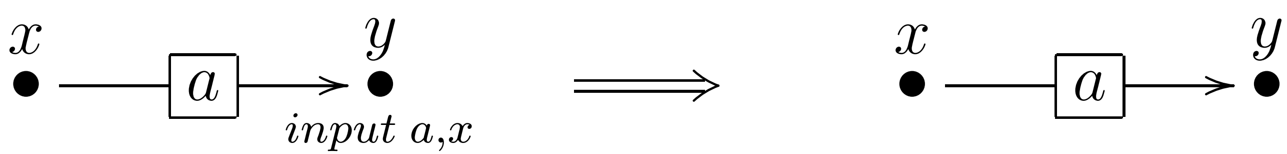
$$x, y \vdash b(x, y) \xrightarrow{\{(x, in\ a, \langle\rangle), (y, \overline{input\ a}, \langle z \rangle)\}} x, y, z \vdash b(x, z)$$

$(input1)$




$$x, y \vdash a(x, y) \xrightarrow{\{(y, input\ a, \langle x \rangle)\}} x, y \vdash a(x, y)$$

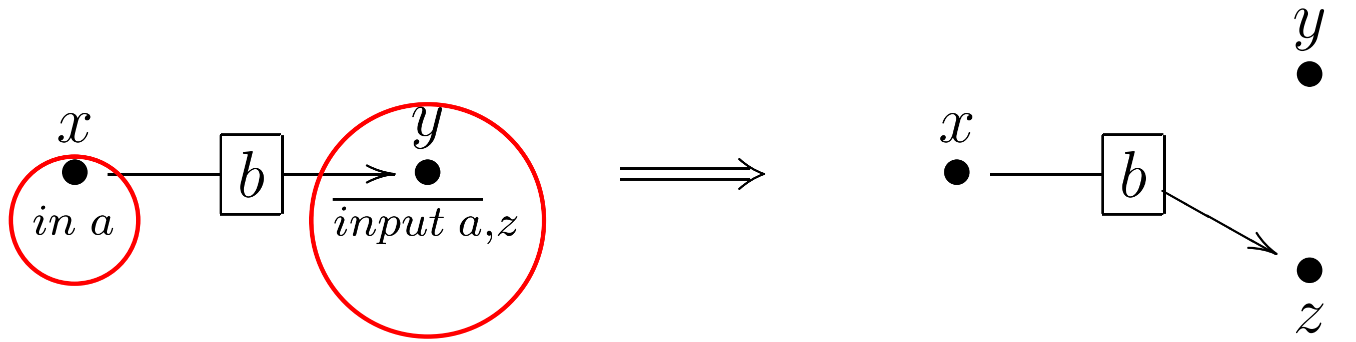
$(input2)$



# Coordination Productions for Ambient

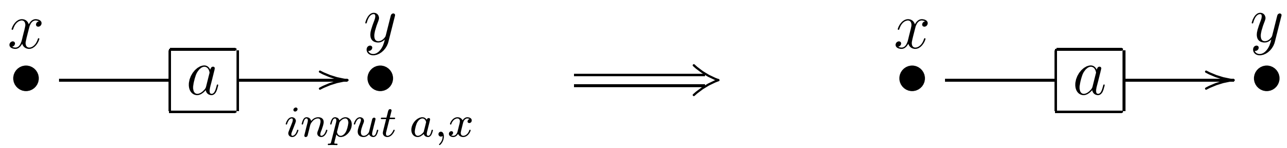
$$x, y \vdash b(x, y) \xrightarrow{\{(x, in\ a, \langle\rangle), (y, \overline{input\ a}, \langle {\color{red}z} \rangle)\}} x, y, {\color{red}z} \vdash b(x, {\color{red}z})$$

$(input1)$




$$x, y \vdash a(x, y) \xrightarrow{\{(y, input\ a, \langle x \rangle)\}} x, y \vdash a(x, y)$$

$(input2)$



# Semantic Correspondence

**Theorem** If $P \rightarrow Q$ then $[\![\, P \,]\!]_x \xrightarrow[id]{\Lambda} [\![\, Q \,]\!]_x$ and

- either $\Lambda = \emptyset$
- or $\Lambda = \{(x, \tau, \langle\rangle)\}$

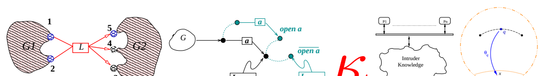
# Semantic Correspondence

**Theorem** If $P \rightarrow Q$ then $[\![\ P\ ]\!]_x \xrightarrow[id]{\Lambda} [\![\ Q\ ]\!]_x$ and

- either $\Lambda = \emptyset$
- or $\Lambda = \{(x, \tau, \langle\rangle)\}$

**Theorem** If $[\![\ P\ ]\!]_x \xrightarrow[\pi]{\Lambda} \Gamma \vdash G$ is a basic transition, then

- either $[\![\ P\ ]\!]_x = \Gamma \vdash G$
- or $\exists Q \in Proc : \quad P \rightarrow Q \ \wedge \ \Gamma \vdash G = [\![\ Q\ ]\!]_x$



# Qlaim: Expressing and reasoning on Connection Properties

**In collaboration with R. De Nicola, G. Ferrari, U. Montanari, R. Pugliese**

# Klaim [?]

# Klaim [?]

- Multiple TS

# Klaim [?]

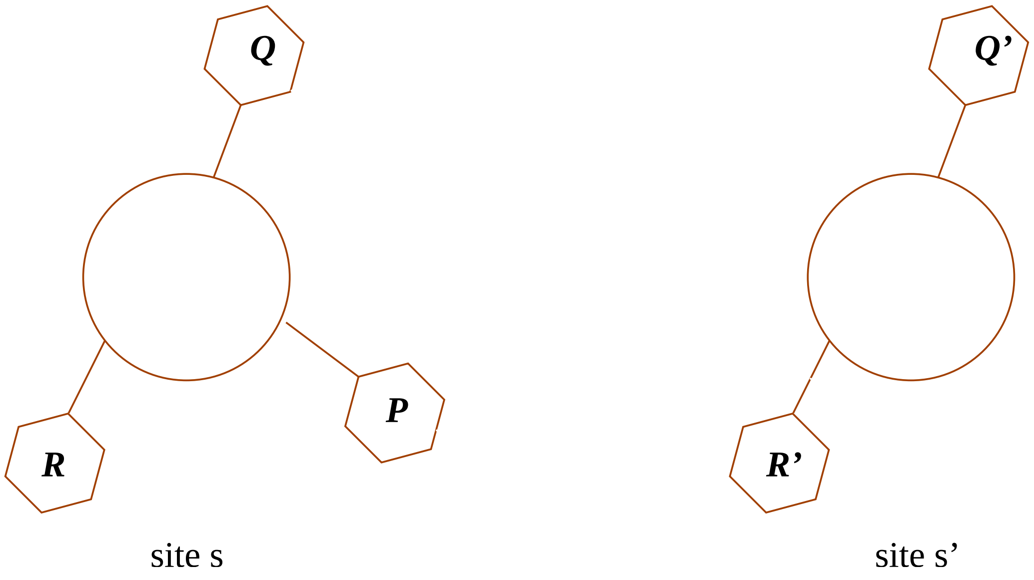
- Multiple TS
- Localities: first class citizens

# Klaim [?]

- Multiple TS
- Localities: fi rst class citizens
- Process migration

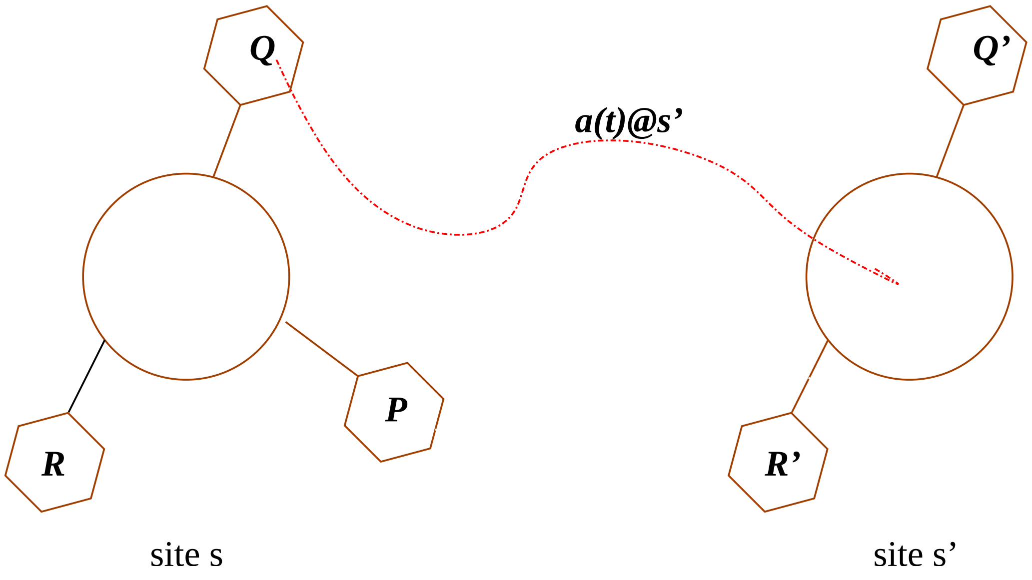
# Klaim [?]

- Multiple TS
- Localities: first class citizens
- Process migration



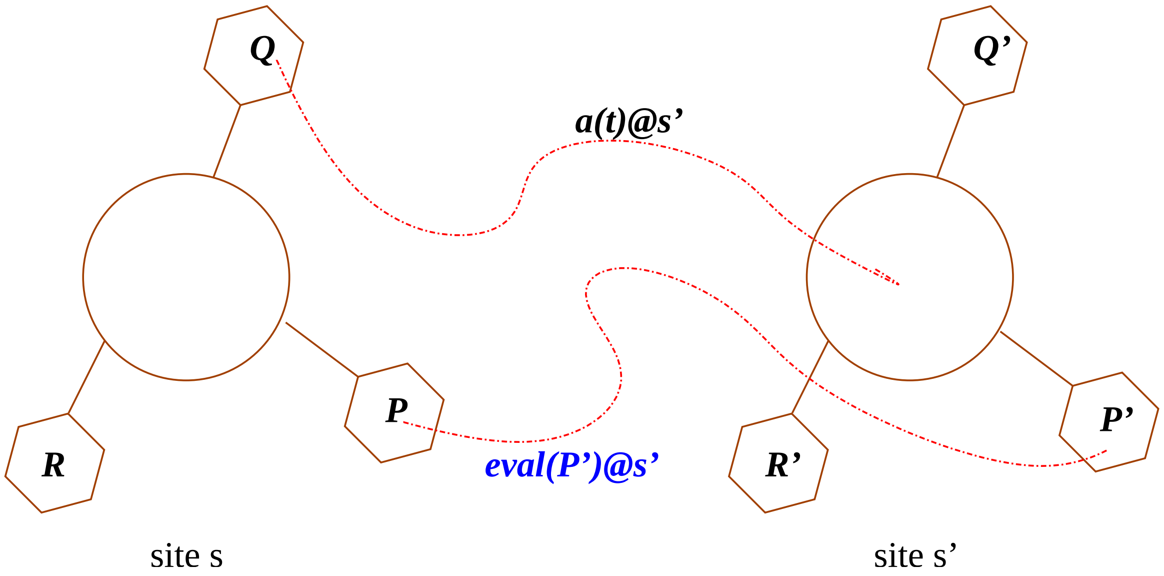
# Klaim [?]

- Multiple TS
- Localities: first class citizens
- Process migration



# Klaim [?]

- Multiple TS
- Localities: first class citizens
- Process migration




$$
\begin{array}{lcl}
P & ::= & \mathbf{nil} \\
 & \mid & \alpha.P \\
 & \mid & P_1 \mid P_2 \\
\alpha & ::= & a@s \\
a & ::= & \ldots \text{ // Klaim actions} \\
 & \mid & \mathbf{eval}(P)
\end{array}
$$

# Qlaim: Gateways

In [?]

# Qlaim: Gateways

In [?]

- Coordinators (super processes)

# Qlaim: Gateways

In [?]

- Coordinators (super processes)
- Dynamic creation of sites

# Qlaim: Gateways

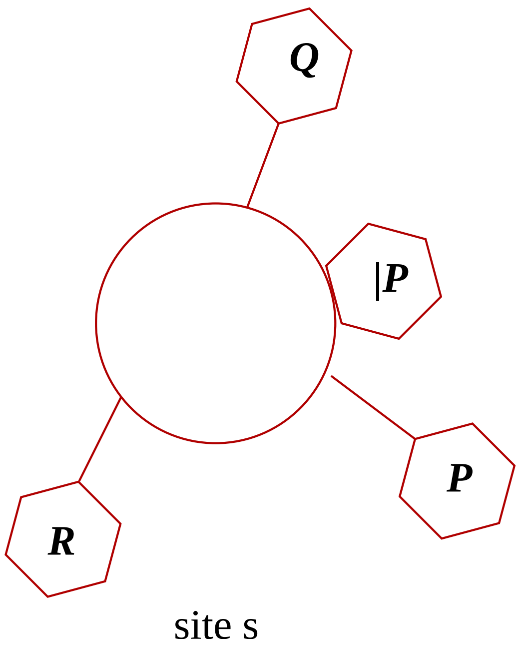
In [?]

- Coordinators (super processes)
- Dynamic creation of sites
- Gateway connection management

# Qlaim: Gateways

In [?]

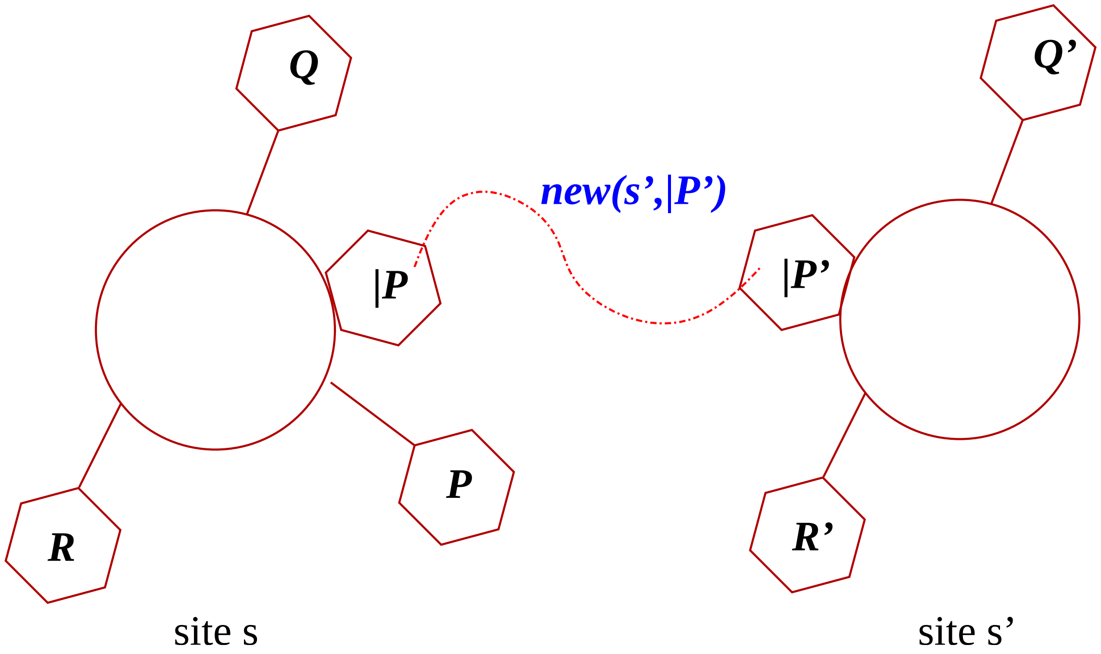
- Coordinators (super processes)
- Dynamic creation of sites
- Gateway connection management



# Qlaim: Gateways

In [?]

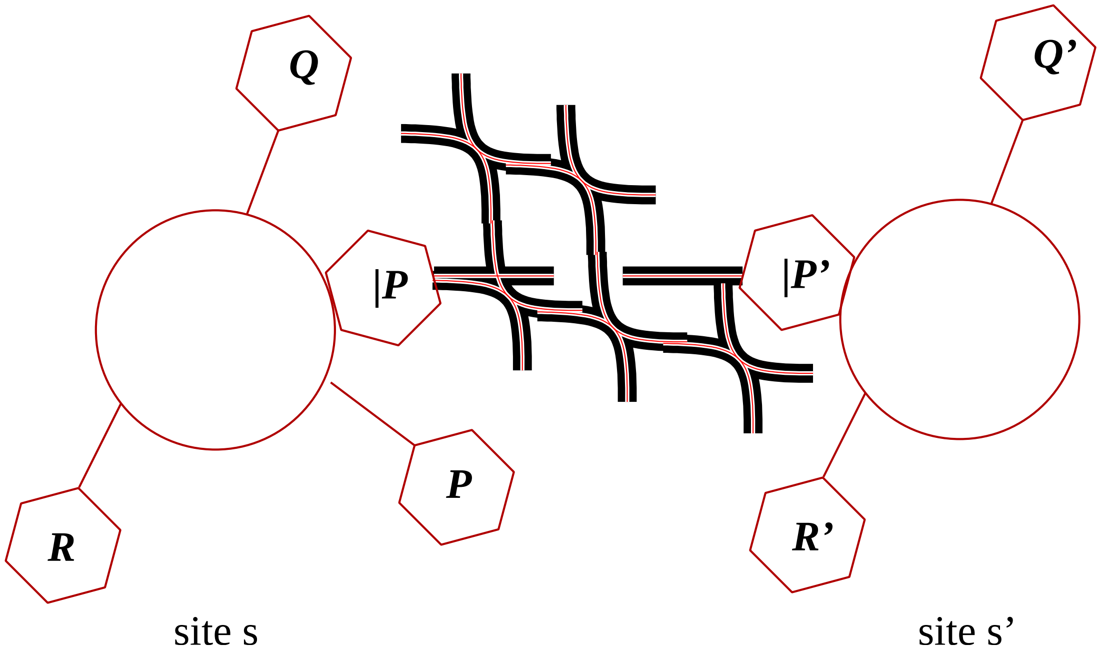
- Coordinators (super processes)
- Dynamic creation of sites
- Gateway connection management



# Qlaim: Gateways

In [?]

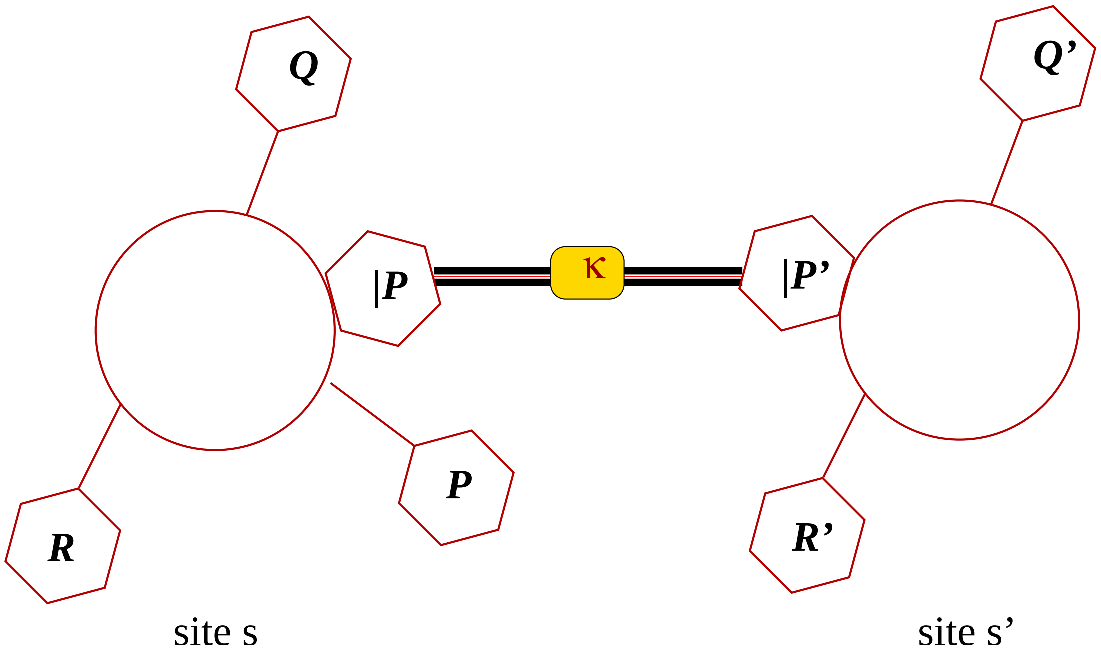
- Coordinators (super processes)
- Dynamic creation of sites
- Gateway connection management



# Qlaim: Gateways

In [?]

- Coordinators (super processes)
- Dynamic creation of sites
- Gateway connection management




$$
\begin{array}{lcl}
\mathbb{P} & ::= & \gamma.\mathbb{P} \mid \mathbb{P}_1 \mid \mathbb{P}_2 \\
\gamma & ::= & \alpha \\
 & \mid & \mathbf{new}(s, \mathbb{P}) \\
 & \mid & \mathbf{login}(s, \kappa) \\
 & \mid & \mathbf{accept}(s, \kappa) \\
 & \mid & \mathbf{logout}(s, \kappa) \\
 & \mid & \mathbf{disconnect}(s, \kappa)
\end{array}
$$

# Connection costs

Cost $\kappa$ abstracts characteristics of connections (bandwidth, latency, distance, access rights ...)

# Connection costs

Cost $\kappa$ abstracts characteristics of connections (bandwidth, latency, distance, access rights ...)

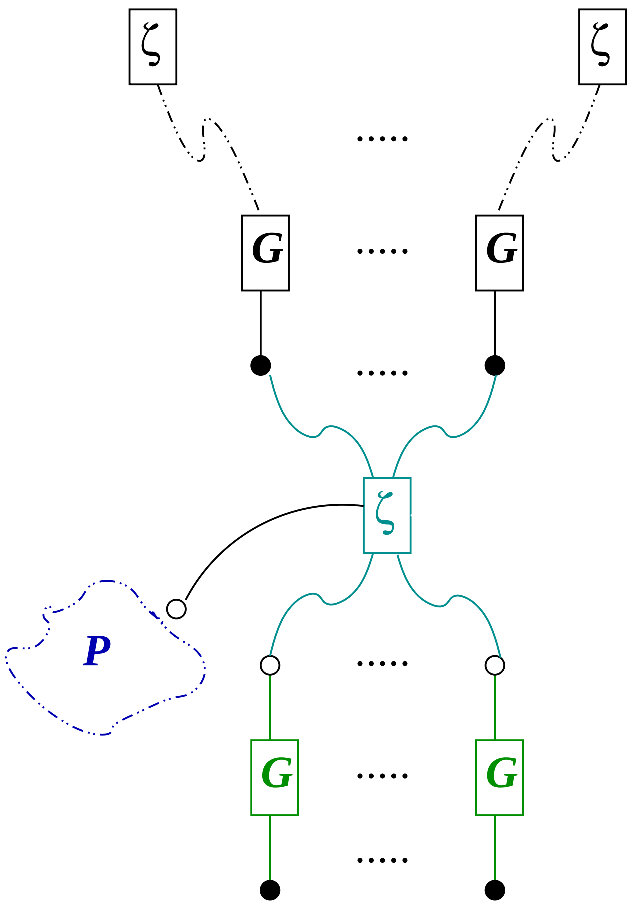
Algebra on costs: c-semiring. For instance

$$\langle c_1, \pi_1 \rangle \oplus \langle c_2, \pi_2 \rangle = \langle c_1 + c_2, \pi_1 \cup \pi_2 \rangle$$

$$\langle c_1, \pi_1 \rangle \otimes \langle c_2, \pi_2 \rangle = \begin{cases} \langle c_1 + c_2, \pi_1 \cap \pi_2 \rangle & \text{if } c_2 < c_1 \text{ and } \pi_2 \subset \pi_1 \\ \bot & \text{otherwise} \end{cases}$$

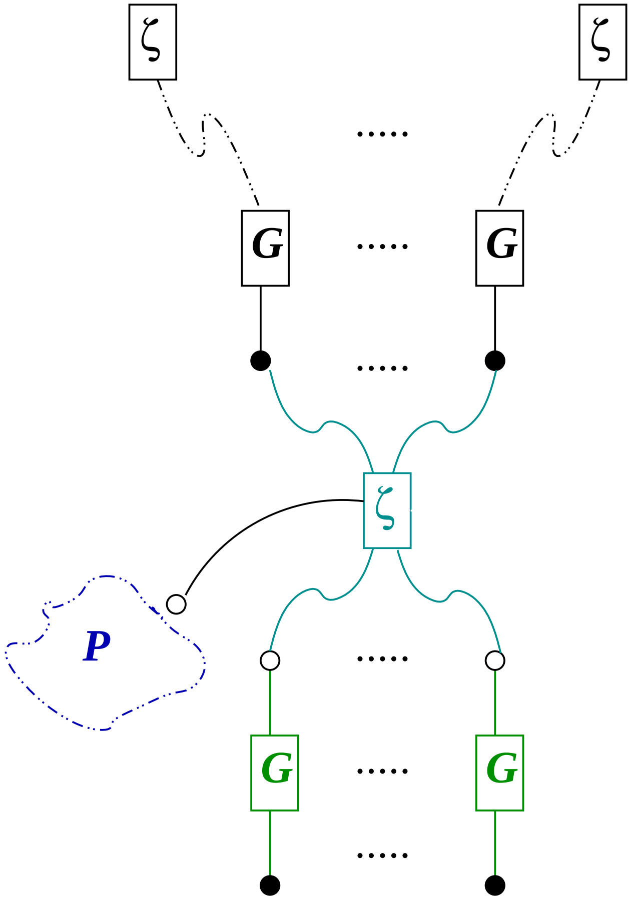
# Qlaim & Hypergraphs

$$[\![ \, s ::^{L,} P \, ]\!] \; = \; \Gamma \vdash (\nu \, \vec{x}, p)([\![ \, P \, ]\!]_p \mid \mathfrak{S}^{s}_{m,n}(\vec{u}, \vec{x}, p) \mid \prod_{j=1}^{n} G^{\kappa_j}_{t_j}(x_j, v_j))$$



# Qlaim & Hypergraphs

$$[\![\, s ::^{L,}\ P \,]\!] \;=\; \Gamma \vdash (\nu\, \vec{x}, p)([\![\, P \,]\!]_p \mid \mathfrak{S}^{s}_{m,n}(\vec{u}, \vec{x}, p) \mid \prod_{j=1}^{n} G^{\kappa_j}_{t_j}(x_j, v_j))$$




$$
\begin{array}{lcl}
[\![\, \mathbf{nil} \,]\!]_p & = & nil \\
[\![\, \mathbf{out}t \,]\!]_p & = & L_{\mathbf{out}t}(p) \\
[\![\, \gamma.P \,]\!]_p & = & L_{\gamma.P}(p) \\
[\![\, \mathbf{eval}(P)@s \,]\!]_p & = & (\nu\, u)(\mathbf{eval}^{T(P)}_{s}(u,p) \mid S_P(u)) \\
[\![\, P_1 \mid P_2 \,]\!]_p & = & [\![\, P_1 \,]\!]_p \mid [\![\, P_2 \,]\!]_p \\
[\![\, rec\, X.\, P \,]\!]_p & = & [\![\, P[^{rec\, X.\, P}/_{X}] \,]\!]_p.
\end{array}
$$

# Qlaim's Graph semantics: pros & cons

# Qlaim’s Graph semantics: pros & cons

– Many productions (recently reduced :-)

# Qlaim’s Graph semantics: pros & cons

– Many productions (recently reduced :-)

+ Determines the “optimal” path (also Qlaim)

# Qlaim’s Graph semantics: pros & cons

- – Many productions (recently reduced :-)
- \+ Determines the “optimal” path (also Qlaim)
- \+ Path reservation

# Qlaim’s Graph semantics: pros & cons

– Many productions (recently reduced :-)

+ Determines the “optimal” path (also Qlaim)

+ Path reservation

+ Path routing

# Qlaim's Graph semantics: pros & cons

- – Many productions (recently reduced :-)
- \+ Determines the "optimal" path (also Qlaim)
- \+ Path reservation
- \+ Path routing

**Theorem** Qlaim remote actions are routed on paths with minimal cost (wrt the c-semiring operations)

# Hypergraph & Software Design

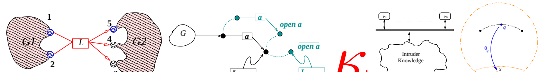
In [?] graph transformation is used for modelling dynamic behaviour of UML specifications.

- \+ Formal semantics of computations
- – No local rewritings
- – Distribution is not considered

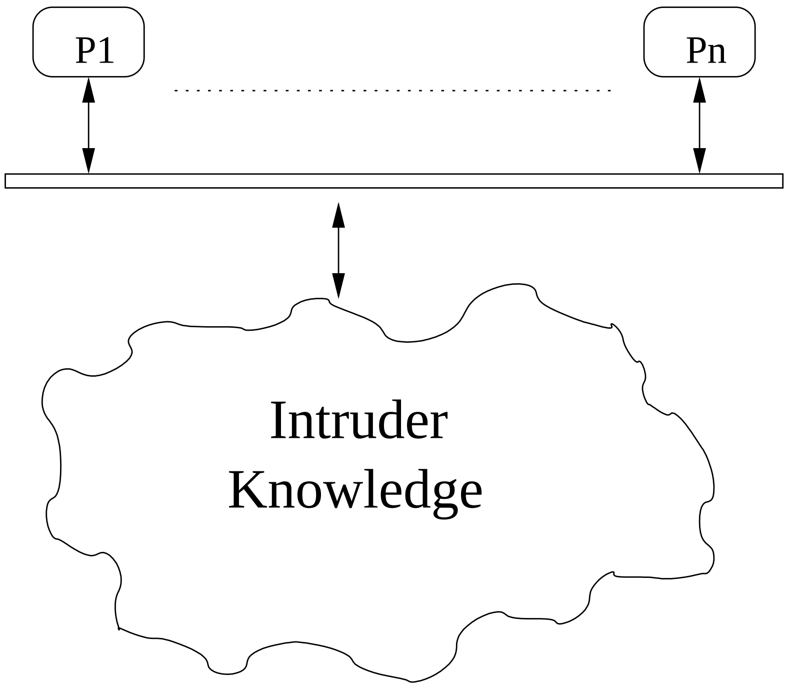
SHR has been applied as a further refinement step in the software design process.

# Security

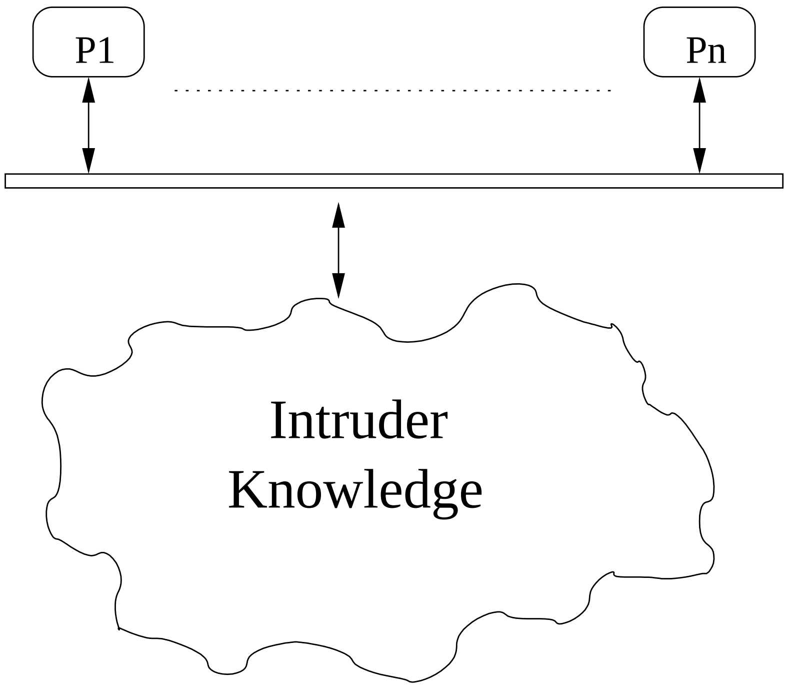
**In collaboration with A. Bracciali, A. Brogi and G. Ferrari**



# The Dolev-Yao Model

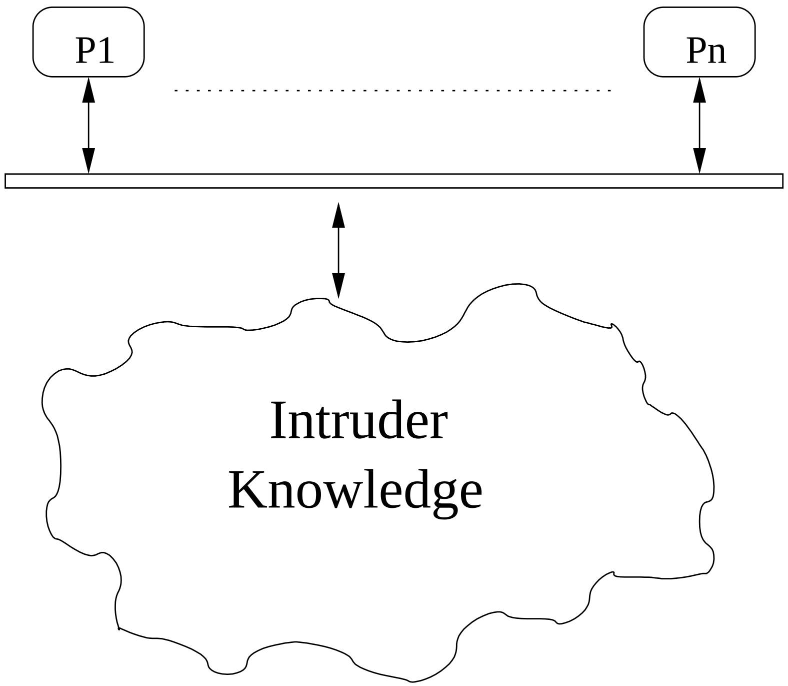


# The Dolev-Yao Model




- Receive and store any transmitted message
- Hinder a message
- Decompose messages into parts
- Forge messages using known data
- Perfect Encryption Hypothesis

# The Dolev-Yao Model




- Receive and store any transmitted message
- Hinder a message
- Decompose messages into parts
- Forge messages using known data
- Perfect Encryption Hypothesis

Names $n, m, ..., A, B, S, ...$

Keys $k, k', ..., A^+, A^-, ...$

Messages $M \quad ::= \quad N \mid K \mid M, M \mid \{M\}_M$

# Intruder capabilities: $\bowtie$

$$\frac{m \in \kappa}{\kappa \bowtie m}\,(\in) \qquad \frac{\kappa \bowtie m \quad \kappa \bowtie n}{\kappa \bowtie m, n}\,(,) \qquad \frac{\kappa \bowtie m \quad \kappa \bowtie \lambda}{\kappa \bowtie \{m\}_\lambda}\,(\{\})$$

$$\frac{\kappa \bowtie m, n}{\kappa \bowtie m}\,(+_1) \qquad \frac{\kappa \bowtie m, n}{\kappa \bowtie n}\,(+_2) \qquad \frac{\kappa \bowtie \{m\}_\lambda \quad \kappa \bowtie \lambda^-}{\kappa \bowtie m}\,(\}\{)$$

Generalising [?] to asymmetric key cryptography

**Theorem** $\bowtie$ is decidable

# A Calculus of Principals

## Some design choices:

- Extension of IP [?]
- Cryptography & communication (pattern-matching)
- Key-sharing via “name fusion”
- Rôle based calculus
- Multi-session facilities

$$
\begin{array}{lcl}
E, F & ::= & \mathbf{nil} \mid \alpha.E \mid E + E \mid E \parallel E \\
\alpha, \beta & ::= & in(d) \mid out(d) \\
d & ::= & N \mid K \mid d, d \mid \{d\}_d \mid x \mid ?x
\end{array}
$$

# A cIP protocol

$$
\begin{array}{l}
1. A \rightarrow B : \{na, A\}_{B^+} \\
2. B \rightarrow A : \{na, nb\}_{A^+} \\
3. A \rightarrow B : \{nb\}_{B^+}
\end{array}
$$

$$
A \triangleq (y)[\; \begin{array}{l} out(\{na, A\}_{y^+}). \\ in(\{na, ?u\}_{A^-}). \\ out(\{u\}_{y^+})] \end{array}
$$

$$
B \triangleq ()[\; \begin{array}{l} in(\{?x, ?z\}_{B^-}). \\ out(\{x, nb\}_{z^+}). \\ in(\{nb\}_{B^-})] \end{array}
$$

# cIP Semantics

$$\frac{}{\alpha.E \xrightarrow{\alpha} E} \qquad \frac{E \xrightarrow{\alpha} E'}{E + F \xrightarrow{\alpha} E'}$$

$$\frac{E \xrightarrow{\alpha} E'}{E \parallel F \xrightarrow{\alpha} E' \parallel F} \; \mathrm{bn}(\alpha) \cap \mathrm{fn}(F) = \emptyset$$

$$\frac{E_i \xrightarrow{in(d)} E'_i \qquad \partial(\kappa) \triangleright m :\; \exists \sigma \text{ grounds.t. } \mathrm{d}\sigma \sim \mathrm{m}}{\langle (\tilde{X}_i)[E_i] \cup \mathcal{C}, \chi, \kappa \rangle \;\mapsto\; \langle (\tilde{X}_i)[E'_i \sigma] \cup \mathcal{C}, \chi\sigma, \kappa \rangle}$$

$$\frac{E_i \xrightarrow{out(m)} E'_i}{\langle (\tilde{X}_i)[E_i] \cup \mathcal{C}, \chi, \kappa \rangle \;\mapsto\; \langle (\tilde{X}_i)[E'_i] \cup \mathcal{C}, \chi, \kappa \cup m \rangle}$$

$$\frac{\mathcal{C}' = join(A_i, \gamma, \mathcal{C}) \qquad A \triangleq (\tilde{X})[E] \qquad i \text{ new}}{\langle \mathcal{C}, \chi, \kappa \rangle \;\mapsto\; \langle \mathcal{C}', \chi\gamma, \kappa \cup \{A_i\} \rangle}$$

# A symbolic cIP trace

$$\langle (y_1)[out(\{na_1, A_1\}_{y_1^+}).in(\{na_1, ?u_1\}_{A_1^-}).out(\{u_1\}_{y_1^+})],\ \varepsilon,\ \{A_1\}\rangle$$

$$\rightsquigarrow_{o\ \{na_1, A_1\}_{y_1^+}}$$

$$\langle (y_1)[in(\{na_1, ?u_1\}_{A_1^-}).out(\{u_1\}_{y_1^+})],\ \varepsilon,\ \{A_1, \{na_1, A_1\}_{y_1^+}\}\rangle$$

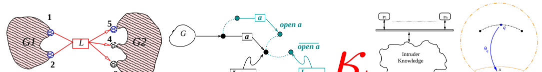
$$\rightsquigarrow_{j\ B_2\ [^{B_2}/_{y_1}]}$$

$$\kappa = \{A_1, B_2, \{na_1, A_1\}_{B_2{}^+}\}$$

$$\left\langle \begin{array}{c} ()[in(\{na_1, ?u_1\}_{A_1^-}).out(\{u_1\}_{B_2{}^+})], \\ ()[in(\{?x_2, ?z_2\}_{B_2^-}).out(\{x_2, nb_2\}_{z_2^+}).in(\{nb_2\}_{B_2^-})] \end{array},\ [^{B_2}/_{y_1}],\ \kappa\right\rangle$$

$$\rightsquigarrow_{i\ \{x_2(\kappa), A_1^+\}_{B_2^+}}$$

$$\left\langle \begin{array}{c} ()[in(\{na_1, ?u_1\}_{A_1^-}).out(\{u_1\}_{B_2^+})], \\ ()[out(\{x_2(\kappa), nb_2\}_{A_1^+}).in(\{nb_2\}_{B_2^-})] \end{array},\ [^{B_2, x_2(\kappa),\ A_1}/_{y_1, x_2, z_2}], \kappa\right\rangle$$



# $\mathcal{PL}$: Formalising Security Properties

$$
\begin{array}{lcl}
\phi, \psi & ::= & \delta \in \mathfrak{K} \\
 & | & \forall \alpha : A.\phi \\
 & | & x@\alpha = \delta \\
 & | & \alpha = \beta \\
 & | & \neg \phi \ | \ \phi \wedge \psi \\
 & & \\
\delta & ::= & d \ | \ \alpha \ | \ x@\alpha
\end{array}
$$

$$\kappa \models_{\chi} \phi$$

*“If $B$ completes a protocol session and thinks that he has been talking to $A$, then $A$ had started a protocol session thinking that she has been talking to $B$”*

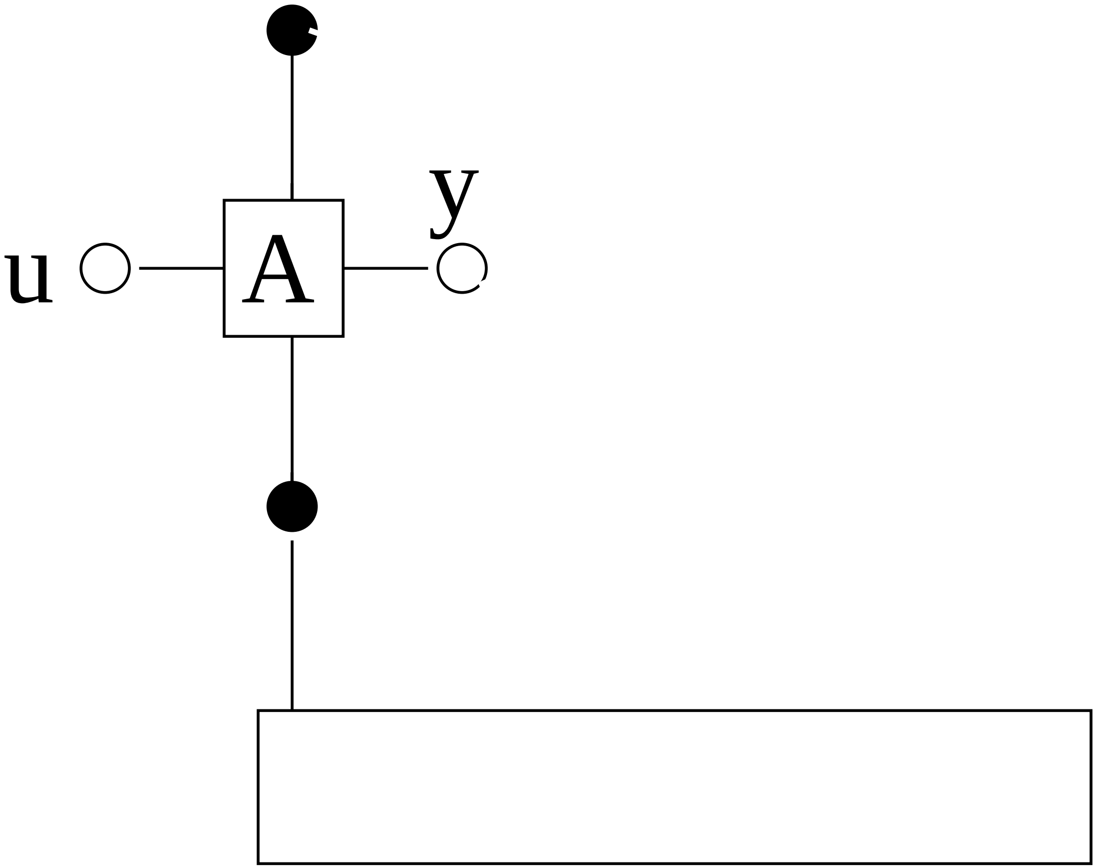
$$\forall \beta : B.\exists \alpha : A.(z@\beta = \alpha \rightarrow y@\alpha = \beta)$$

# Hypergraphs for security

$$A \triangleq (y)[\quad out(\{na, A\}_{y^+}).$$
$$in(\{na, ?u\}_{A^-}).$$
$$out(\{u\}_{y^+})]$$

$$B \triangleq ()[\quad in(\{?x, ?z\}_{B^-}).$$
$$out(\{x, nb\}_{z^+}).$$
$$in(\{nb\}_{B^-})]$$

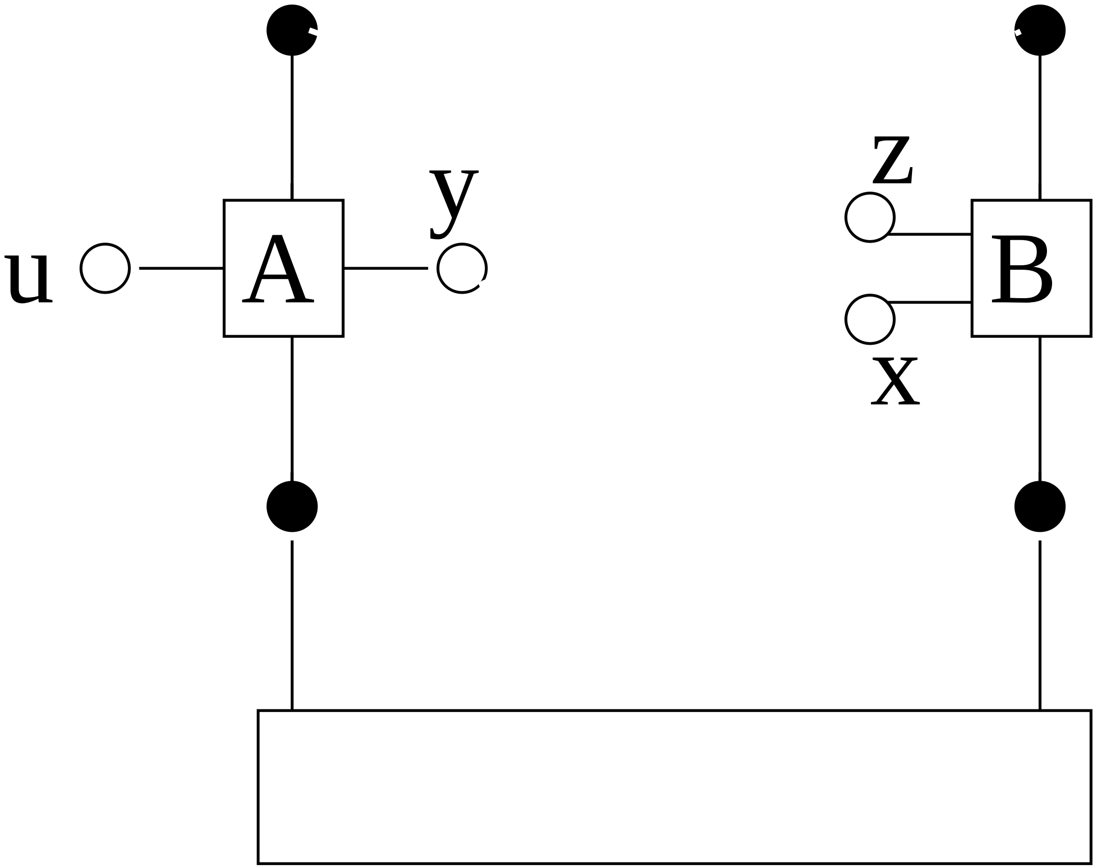
# Hypergraphs for security




$$A \stackrel{\triangle}{=} (y)[\quad out(\{na, A\}_{y^+}).$$
$$in(\{na, ?u\}_{A^-}).$$
$$out(\{u\}_{y^+})]$$

$$B \stackrel{\triangle}{=} ()[\quad in(\{?x, ?z\}_{B^-}).$$
$$out(\{x, nb\}_{z^+}).$$
$$in(\{nb\}_{B^-})]$$

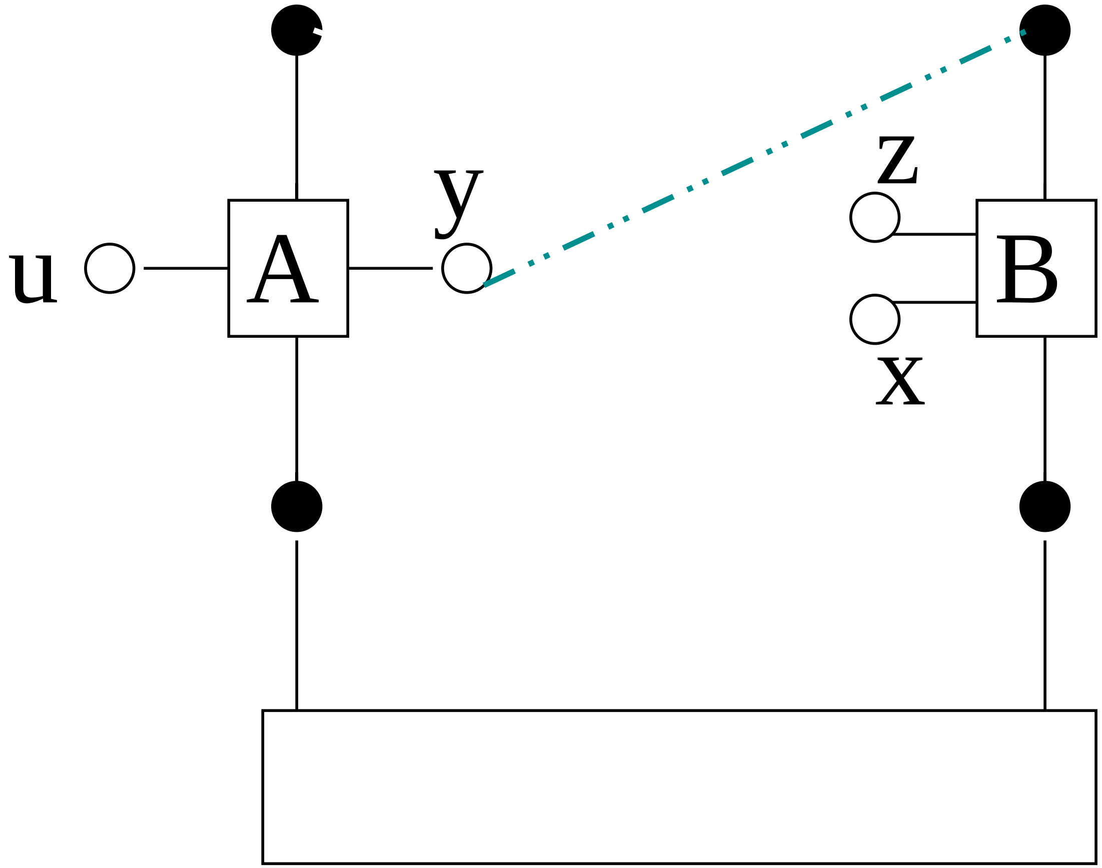
# Hypergraphs for security




$$A \triangleq (y)[\quad out(\{na, A\}_{y^+}).$$
$$in(\{na, ?u\}_{A^-}).$$
$$out(\{u\}_{y^+})]$$

$$B \triangleq ()[\quad in(\{?x, ?z\}_{B^-}).$$
$$out(\{x, nb\}_{z^+}).$$
$$in(\{nb\}_{B^-})]$$

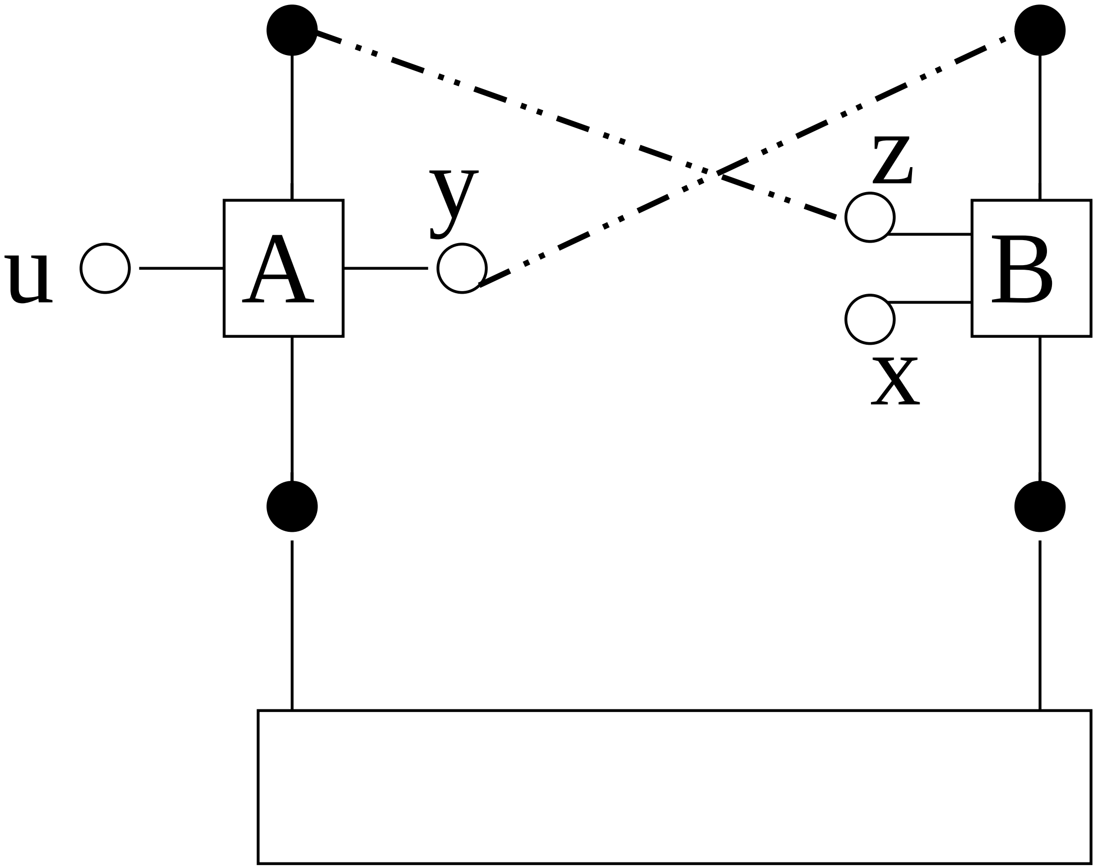
# Hypergraphs for security




$$A \triangleq (y)[\quad out(\{na, A\}_{y^+}).\ in(\{na, ?u\}_{A^-}).\ out(\{u\}_{y^+})]$$

$$B \triangleq ()[\quad in(\{?x, ?z\}_{B^-}).\ out(\{x, nb\}_{z^+}).\ in(\{nb\}_{B^-})]$$

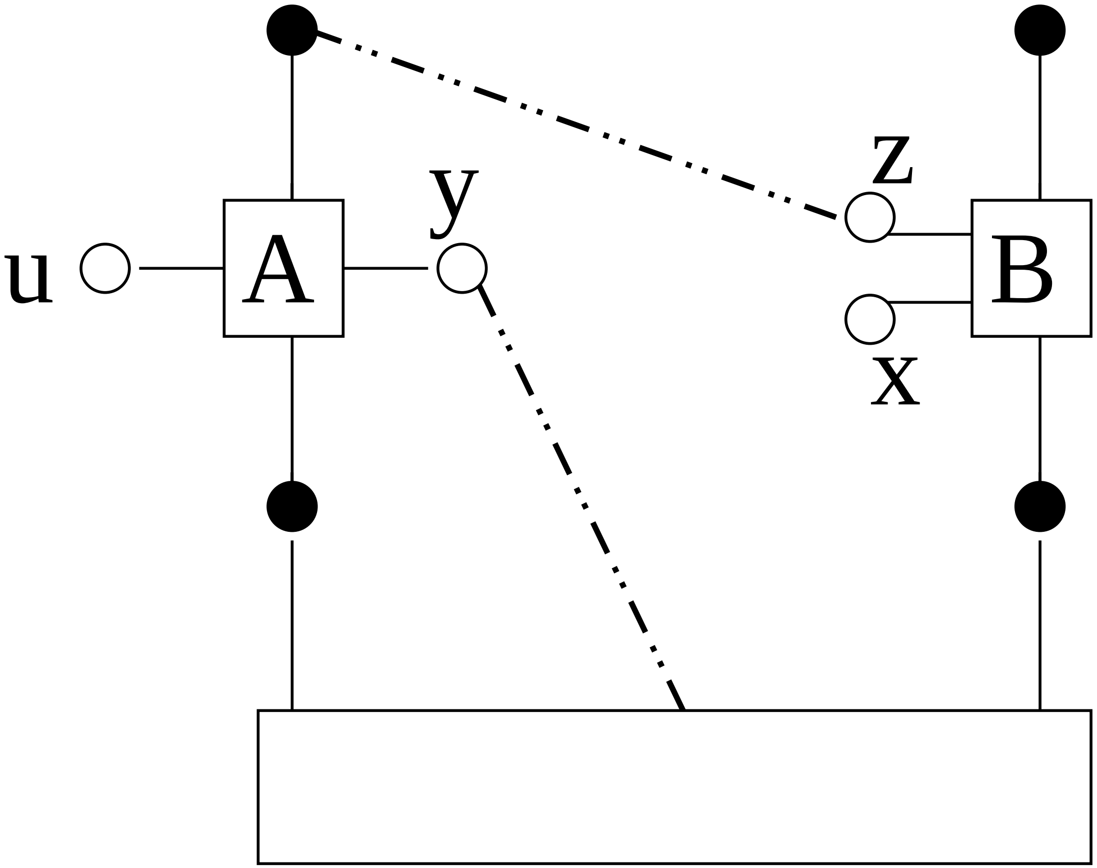
# Hypergraphs for security




$$A \stackrel{\triangle}{=} (y)[\quad out(\{na, A\}_{y^+}).$$
$$in(\{na, ?u\}_{A^-}).$$
$$out(\{u\}_{y^+})]$$

$$B \stackrel{\triangle}{=} ()[\quad in(\{?x, ?z\}_{B^-}).$$
$$out(\{x, nb\}_{z^+}).$$
$$in(\{nb\}_{B^-})]$$

# Hypergraphs for security




$$A \stackrel{\triangle}{=} (y)[\quad out(\{na, A\}_{y^+}).$$
$$in(\{na, ?u\}_{A^-}).$$
$$out(\{u\}_{y^+})]$$

$$B \stackrel{\triangle}{=} ()[\quad in(\{?x, ?z\}_{B^-}).$$
$$out(\{x, nb\}_{z^+}).$$
$$in(\{nb\}_{B^-})]$$

# Mihda: Co-Algebraic Minimisation of Automata

**In collaboration with G. Ferrari, U. Montanari and R. Raggi**

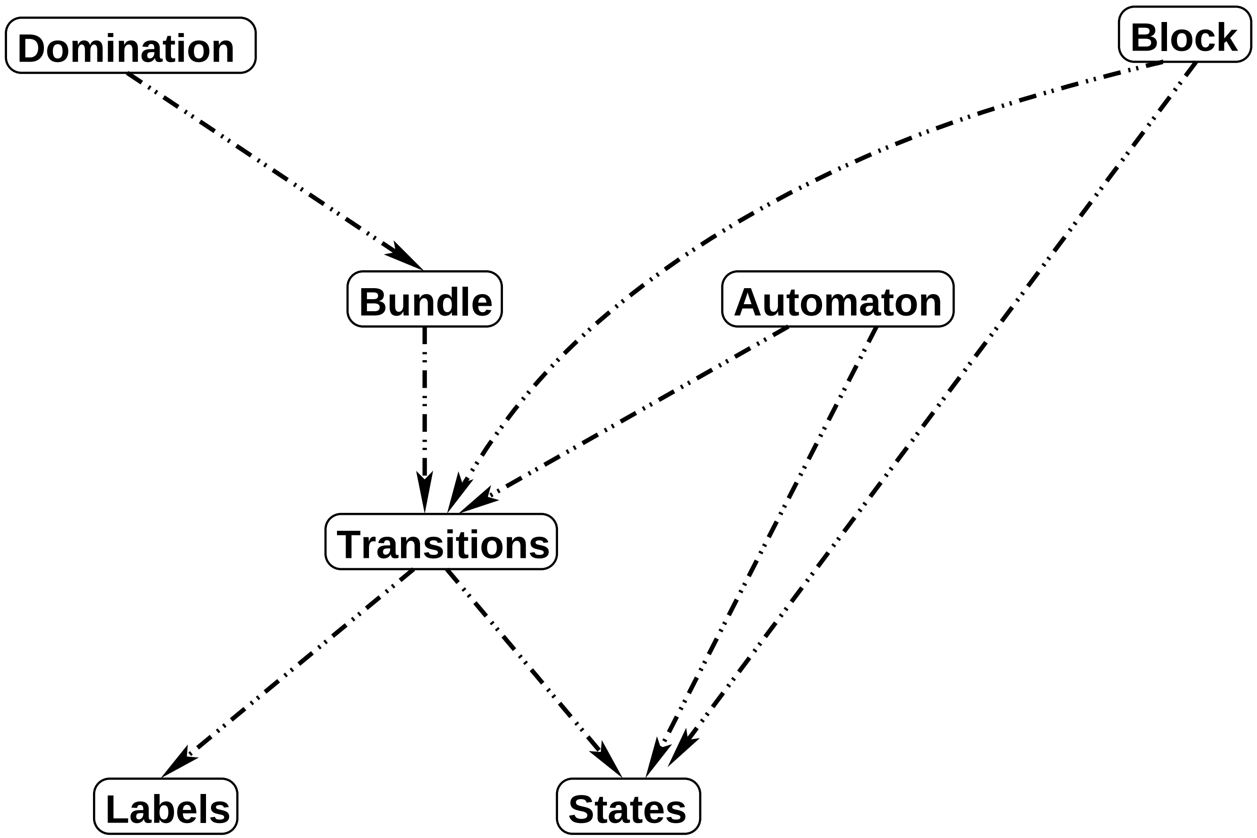
# Mihda

Minimizing History Dependent Automata:

- HD-automata for history dependent calculi
- Co-algebraic specification
- Partition Refinement Algorithm based on co-algebraic specification[?]
- Mihda: Ocaml implementation (refining $\lambda^{\rightarrow,\Pi,\Sigma}$ spec.)

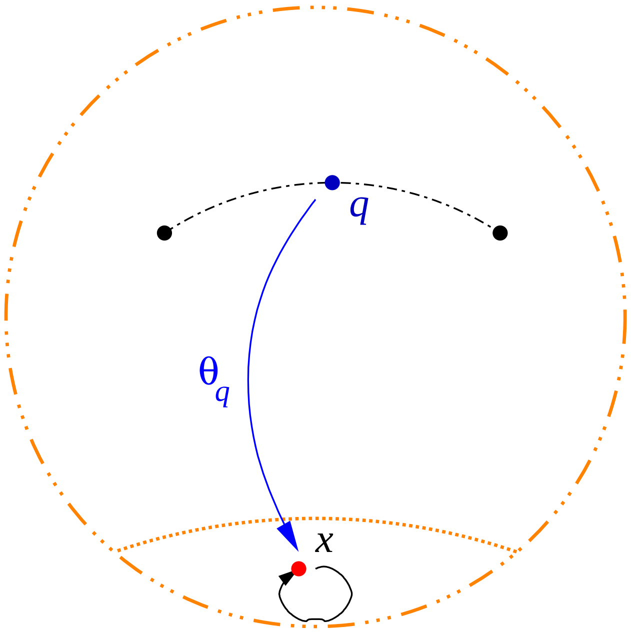
| | Comp. Time | States | Trans. | Min. Time | States | Trans. |
|---|---|---|---|---|---|---|
| GSM small | 0m 0.931s | 211 | 398 | 0m 4.193s | 105 | 197 |
| GSM full | 0m 8.186s | 964 | 1778 | 0m 54.690s | 137 | 253 |

# Mihda Architecture

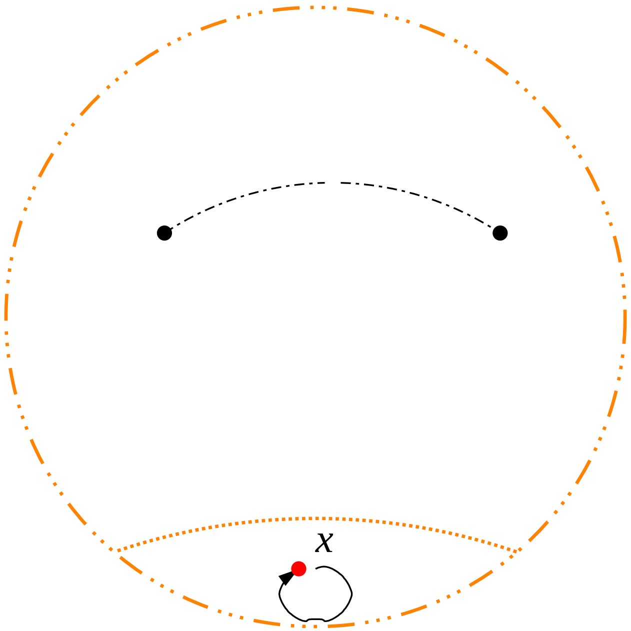


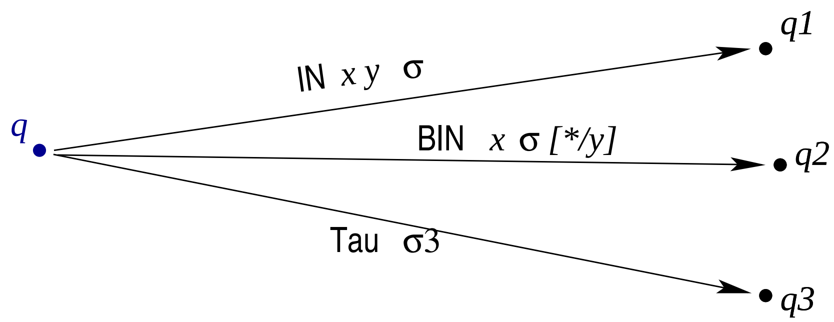

- Adherent to specs
- Highly modular
- Easily extendible

# The main step



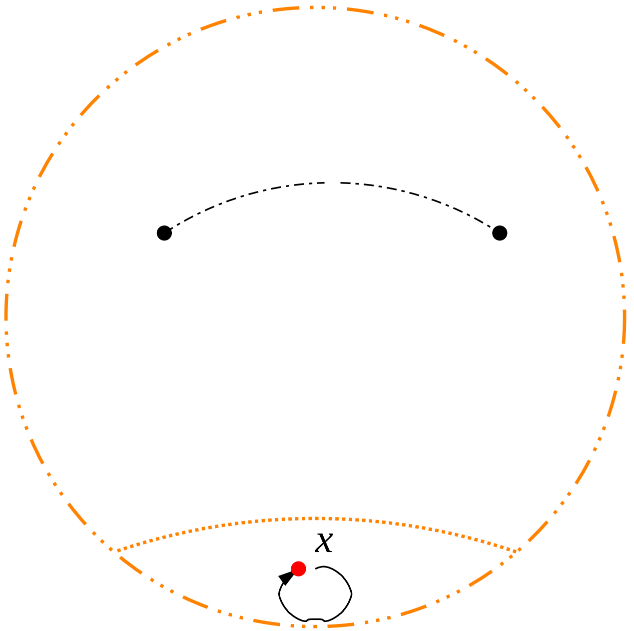
# The main step

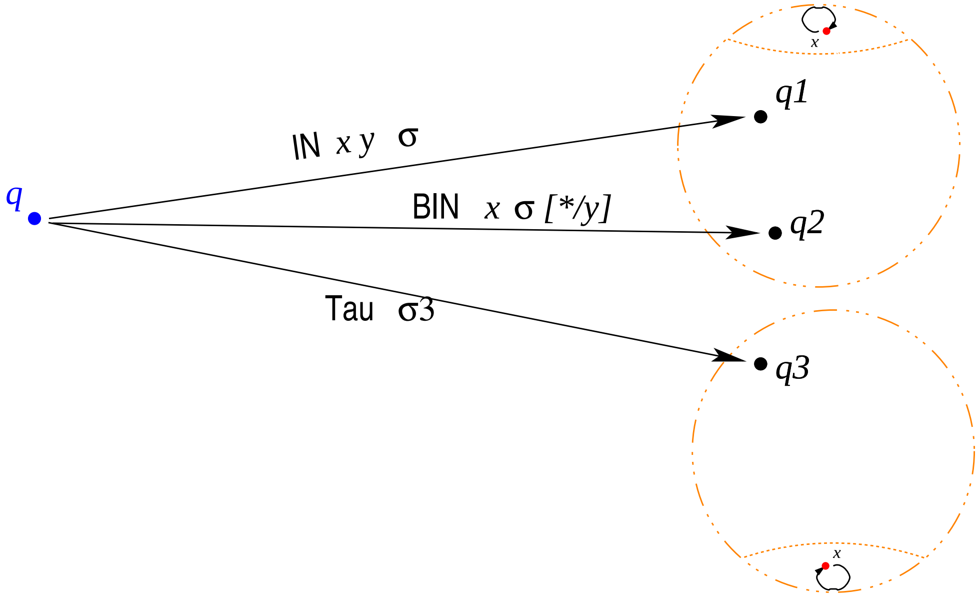



```
let bundle hd q =
  List.sort compare
    (List.filter (fun h → (Arrow.source h) = q) (arrows hd))
```

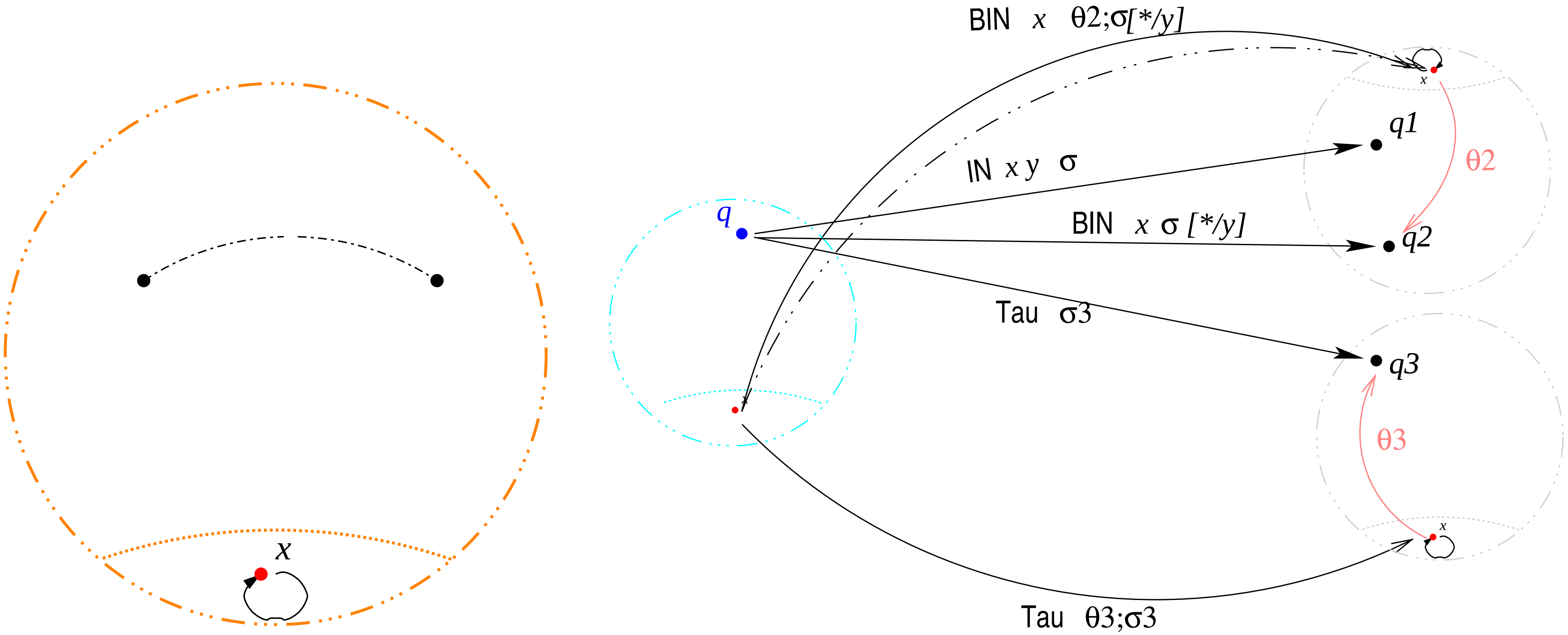
# The main step




List.map $h_n$ bundle

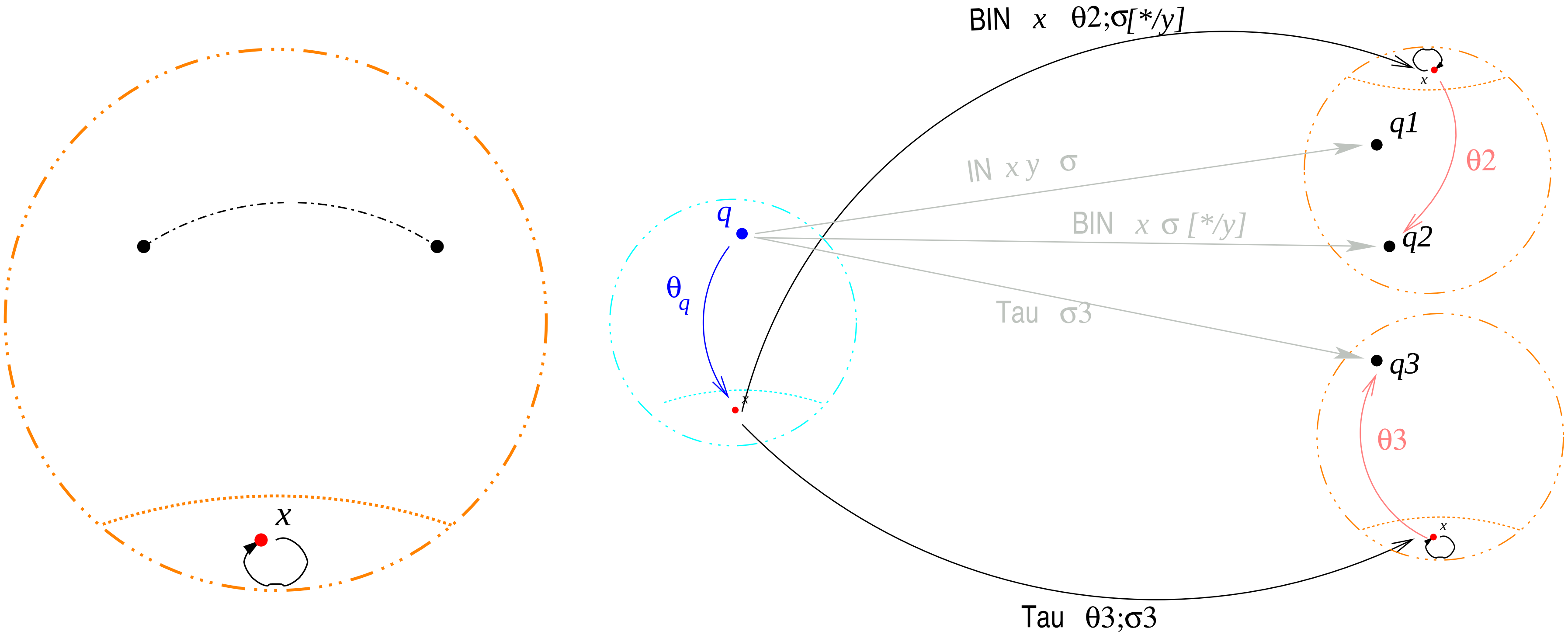
# The main step




$$h_{n+1} \;=\; norm\langle \text{states}, \{\langle \ell, \pi, h_n(q'), \sigma'; \sigma\rangle \mid q \xrightarrow{\ell\ \pi\ \sigma} q' \;\wedge\; \sigma' \in \Sigma_n(q')\}\rangle$$

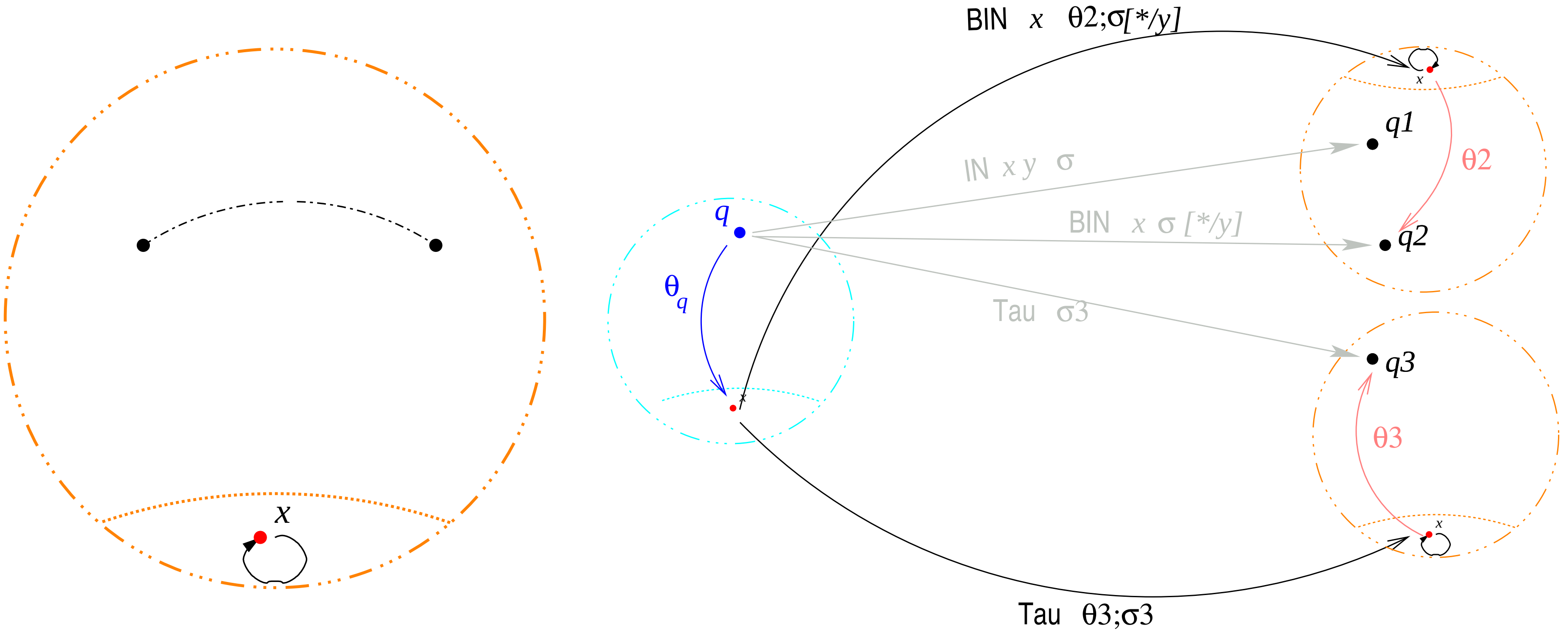
```
let red bl = ......
  let bl_in = List.filter covered_in bl
  in list_diff bl bl_in
```

# The main step



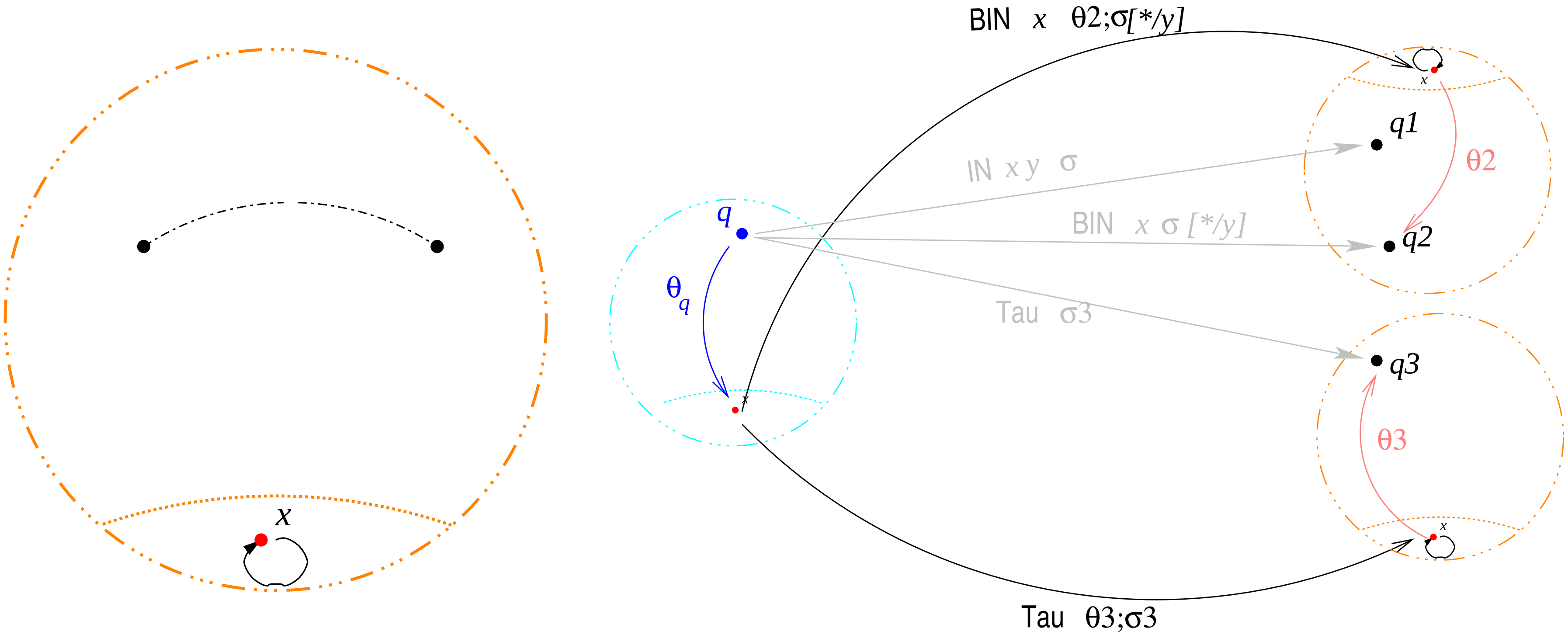

```
let an = active_names_bundle (red bundle) in
let remove_in ar = match ar with
   | Arrow(_,_,In(_,_)) → not (List.mem (obj ar) an)
   | _ → false in
    list_diff bundle (List.filter remove_in bundle)
```

# The main step




$$\Sigma_{n+1}(q) = (\text{compute\_group} \;\; (\text{norm} \;\; \textsf{bundle})) \quad ; \quad \theta_q^{-1}$$

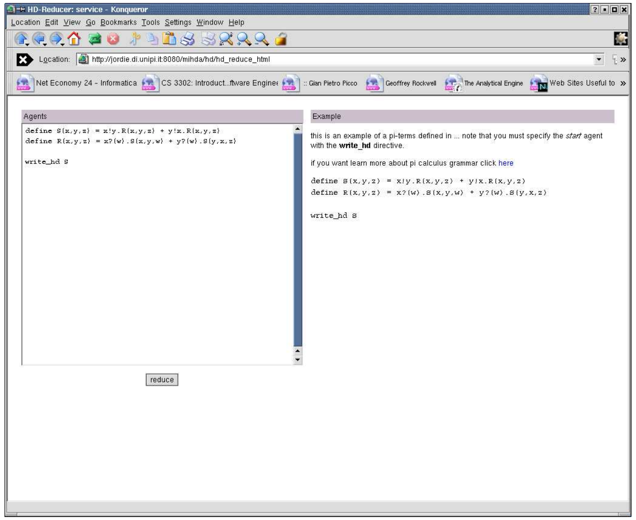
# The main step

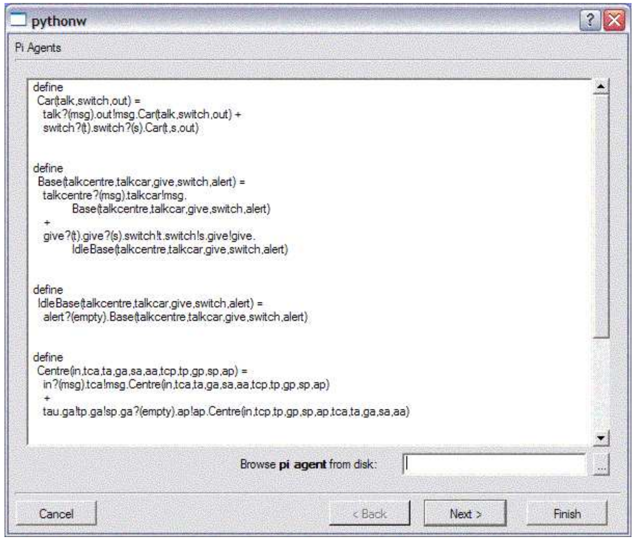


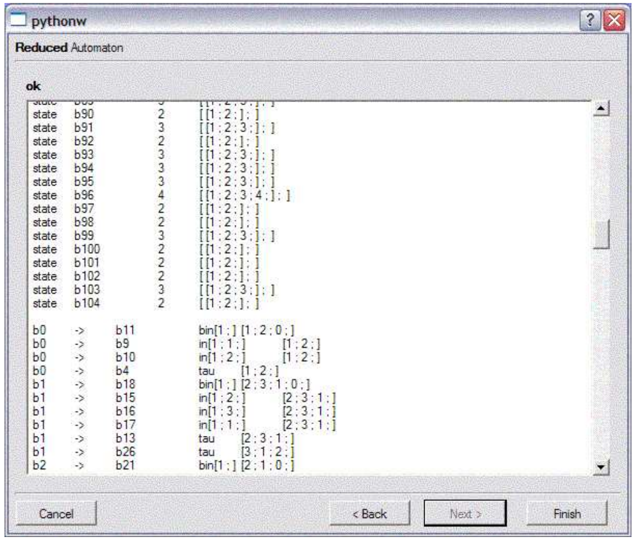

$$\Sigma_{n+1}(q) = (\text{compute\_group}\ \ (\text{norm}\ \ \textsf{bundle}))\quad ;\quad \theta_q^{-1}$$

**Theorem** At the end of each iteration $i$ `blocks` corresponds to $h_{H_i}$

# Mihda Web Interface




http://jordie.di.unipi.it:8080/pweb

# Summing up...

- Declarative approach to WAN programming
  - Foundational aspects
  - QoS at application level
  - Modelling wireless communications (ongoing)
  - Integrating Milner & Hoare synchronizations
  - Web Services
  - Secure composition of components
  - Coordination mechanism
- Tool development
  - Distributed infrastructure
  - Base on Web Services metaphor
  - Proof strategies as programmable coordinators

# Published papers

- Ferrari, G., Pugliese, R., Tuosto, E. Calculi for Network Aware Programming. WOA'00
- Ferrari, G., Montanari, U., Tuosto, E. LTS Semantics of Ambients via Graph Synchronization with Mobility. *ICTCS'01*
- Bracciali, A., Brogi, A., Ferrari, G., Tuosto, E.. Security Issues in Component Based Design. *ConCoord'01*
- Bracciali, A., Brogi, A., Ferrari, G., Tuosto, E., Security and Dynamic Compositions of Open Systems. *PDPTA'02*
- Ferrari, G., Montanari, U., Tuosto, E. Graph-based Models of Internetworking Systems. *UNU/IIST Colloquium'03*
- R. De Nicola, G. Ferrari, Ugo Montanari, R. Pugliese A Formal Basis for Reasoning on Programmable QoS. *International Symposium on Verification'03*

# References

[BBT01] Andrea Bracciali, Antonio Brogi, and Franco Turini. Coordinating interaction patterns. In *Proceedings of the* ACM Symposium on Applied Computing*, Las Vegas, USA*. ACM, 2001.

[BC99] Boumediene Bal, Henri E. Belkhouche and Luca Cardelli, editors. *Workshop on Internet Programming Languages*, volume 1686 of *LNCS*. Springer, 1999.

[BCC01] Michele Bugliesi, Giuseppe Castagna, and Silvia Crafa. Boxed Ambients. In *TACS 2001*, number 2215 in Lecture Notes in Computer Science, pages 38–63. Springer, 2001.

[BLP02] Lorenzo Bettini, Michele Loreti, and Rosario Pugliese. Infrastructure language for open nets. In Proc. of the 2002 ACM Symposium on Applied Computing (SAC'02), Special Track on Coordination Models, Languages and Applications. ACM Press, 2002. Special Track on Coordination Models, Languages and Applications.

[CG00] Luca Cardelli and Andrew D. Gordon. Mobile ambients. *TCS: Theoretical Computer Science*, 240, 2000.

[CJM98] Edmund M. Clarke, Somesh Jha, and Wilfredo R. Marrero. Using state space exploration and a natural deduction style message derivation engine to verify security protocols. In *IFIP Working Conference on Programming Concepts and Methods (PROCOMET)*, 1998.

[CM83] Ilaria Castellani and Ugo Montanari. Graph Grammars for Distributed Systems. In Hartmut Ehrig, Manfred Nagl, and Grzegorz Rozenberg, editors, *Proc. 2nd Int. Workshop on Graph-Grammars and Their Application to Computer Science*, volume 153 of *Lecture Notes in Computer Science*, pages 20–38. Springer-Verlag, 1983.

[DFP98] Rocco De Nicola, Gianluigi Ferrari, and Rosario Pugliese. KLAIM: A kernel language for agents interaction and mobility. *IEEE Transactions on Software Engineering*, 24(5):315–330, 1998.

[DFPV00] Rocco De Nicola, Gianluigi Ferrari, Rosario Pugliese, and Betti Venneri. Types for access control. *Theoretical Computer Science*, 240(1):215–254, June 2000.

[DM87] Pierpaolo Degano and Ugo Montanari. A model of distributed systems based of graph rewriting. *Journal of the ACM*, 34:411–449, 1987.

[FG96] Cedric Fournet and George Gonthier. The reflexive CHAM and the join-calculus. In *Conference Record of POPL '96: The* $23^{\mathrm{rd}}$ *ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 372–385, St. Petersburg Beach, Florida, January 1996.

[FGL$^{+}$96] Cedric Fournet, Georges Gonthier, Jean-Jacques Lévy, Luc Maranget, and Didier Rémy. A calculus of mobile processes. In Ugo Montanari and Vladimiro Sassone, editors, *CONCUR '96: Concurrency Theory, 7th International Conference*, volume 1119 of *Lecture Notes in Computer Science*, pages 406–421, Pisa, Italy, August 1996. Springer-Verlag.

[FMP02] Gianluigi Ferrari, Ugo Montanari, and Marco Pistore. Minimizing transition systems for name passing calculi: A co-algebraic formulation. In Mogens Nielsen and Uffe Engberg, editors, *FOS-*

*SACS 2002*, volume LNCS 2303, pages 129–143. Springer Verlag, 2002.

[HIM00] Dan Hirsch, Paola Inverardi, and Ugo Montanari. Reconfiguration of Software Architecture Styles with Name Mobility. In Antonio Porto and Gruia-Catalin Roman, editors, *Coordination 2000*, volume 1906 of *Lecture Notes in Computer Science*, pages 148–163. Springer Verlag, 2000.

[HM01] Dan Hirsch and Ugo Montanari. Synchronized hyperedge replacement with name mobility: A graphical calculus for name mobility. In *12th International Conference in Concurrency Theory (CONCUR 2001)*, volume 2154 of *LNCS*, pages 121–136, Aalborg, Denmark, 2001. Springer Verlag.

[HR98] Mattew Hennessy and James Riely. Resource access control in systems of mobile agents. In Uwe Nestmann and Benjamin C. Pierce, editors, *HLCL '98: High-Level Concurrent Languages (Nice, France, September 12, 1998)*, volume 16.3 of *entcs*, pages 3–17. Elsevier Science Publishers, 1998. Full version as CogSci Report 2/98, University of Sussex, Brighton.

[HR00] Matthew Hennessy and James Riely. Information flow vs. resource access in the asynchronous pi-calculus. In *27th International Colloquium on Automata, Languages and Programming (ICALP '2000)*, July 2000. A longer version appeared as Computer Science Technical Report 2000:03, School of Cognitive and Computing Sciences, University of Sussex.

[KGKK02] Sabine Kuske, Martin Gogolla, Ralf Kollmann, and Hans-Jörg Kreowski. An Integrated Semantics for UML Class, Object, and State Diagrams based on Graph Transformation. In Michael Butler and Kaisa Sere, editors, *3rd Int. Conf. Integrated Formal Methods (IFM'02)*, Lecture Notes in Computer Science. Springer, Berlin, 2002.

[LS00] Francesca Levi and Davide Sangiorgi. Controlling interference in ambients. In *Conference Record of POPL'00: The 27th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 352–364, Boston, Massachusetts, January 2000.

[MPW92] Robin Milner, Joachim Parrow, and David Walker. A calculus of mobile processes, I and II. *Information* *and Computation*, 100(1):1–40,41–77, September 1992.

[MR96] Ugo Montanari and Francesca Rossi. Graph rewriting and constraint solving for modelling distributed systems with synchronization. In P. Ciancarini and C. Hankin, editors, *Proceedings of the First International Conference COORDINATION '96, Cesena, Italy*, volume 1061 of *LNCS*. Springer Verlag, April 1996.

[VC98] Jan Vitek and Giuseppe Castagna. Towards a calculus of secure mobile computations. In *[**?**]*, Chicago, Illinois, May 1998.