

# Abstract Interpretation of the $\pi$ -Calculus

Arnaud Venet\*

LIX, École Polytechnique, 91128 Palaiseau, France.  
venet@lix.polytechnique.fr  
<http://lix.polytechnique.fr/~venet>

**Abstract.** We are concerned with the static analysis of the communication topology for systems of mobile processes. For this purpose we construct an abstract interpretation of a large fragment of the  $\pi$ -calculus which can be used as a metalanguage to specify the behaviour of these systems. The abstract domain is expressive enough to give accurate descriptions of infinite and non-uniform distributions of processes and communication channels. We design appropriate widening operators for the automatic inference of such information.

## 1 Introduction

The static analysis of communicating processes with dynamically changing structure - commonly called *mobile processes* - has been mostly studied in the particular case of CML-like programs [NN94, Col95a, Col95b]. The design of these analyses heavily depends on type and control-flow information that is specific to CML. In this paper we propose an analysis of communications in the  $\pi$ -calculus [Mil91, MPW92]. Our choice is motivated by the fact that the  $\pi$ -calculus is a widely accepted model for describing mobile processes. It is fairly simple and yet very expressive, since it can encode data structures [Mil91], higher-order communication [Mil91, San94a] and various  $\lambda$ -calculi [Mil92].

Our purpose is to analyze the *communication topology* of a system of mobile processes, i.e. the distribution of processes and communication channels during the evolution of the system. In the standard semantics of the  $\pi$ -calculus this information is encoded within a process algebra. We construct a refinement of this semantics where an instance of a process at a certain stage of evolution of the system is represented by the sequence of internal computations that lead to it. The communication channels are represented in turn as the congruence classes of an equivalence relation over the communication ports of all process instances. This semantic model originated in the study of data structures [Jon81] and has been successfully applied to the alias analysis of ML-like programs by A. Deutsch [Deu92a, Deu92b, Deu94] who designed a powerful analysis based on Abstract Interpretation [CC77, CC92].

C. Colby [Col95b] further extended Deutsch's work and built an analysis for a subset of CML which can discover non-uniform descriptions of infinitely growing communication topologies. However, the whole framework relies on having a

---

\* This work was partly supported by ESPRIT BRA 8130 LOMAPS.

good approximation of the control-flow of the program before doing the analysis of communications. Whereas a closure analysis can be used for higher-order CML programs, there is no realistic solution for the  $\pi$ -calculus where the *only* kind of computation is communication. Therefore we design a new abstract interpretation which gets rid of this problem and still ensures a comparable level of accuracy. We use the technique of *cofibered domains* introduced in [Ven96] which allows us to infer *simultaneously* an approximation of control-flow and communications.

The paper is organized as follows. In Sect. 2 we present the syntax and semantics of the  $\pi$ -calculus. The refined semantics is described in Sect. 3. In Sect. 4 we present the basic concepts of Abstract Interpretation. We construct an abstract domain for the analysis of the  $\pi$ -calculus in Sect. 5. The abstract semantics of the  $\pi$ -calculus is described in Sect. 6. In Sect. 7 we design widening operators in order to make the analysis effective.

## 2 The $\pi$ -Calculus

The basic entities in the  $\pi$ -calculus are *names*, which are provided to represent communication channels. The key point is that computation is restricted to the transmission and reception of names along channels. There are various possibilities of defining processes in the  $\pi$ -calculus, depending on the way communication, recursion and nondeterminism are handled. Our presentation is based on a subset of the *polyadic  $\pi$ -calculus* [Mil91]. Let  $\mathcal{X} = \{x, y, \dots\}$  be an infinite set of names. The syntax of  $\pi$ -terms  $\mathcal{P} = \{P, Q, \dots\}$  is given by the following grammar:

$P ::= \sum_{i \in I} \pi_i.P_i$	guarded sum of finitely many processes
$P \mid Q$	parallel composition
$\nu x.P$	restriction
$!u.P$	guarded replication
$\pi ::= \iota \mid o$	atomic action
$\iota ::= x(\mathbf{y})$	input
$o ::= \bar{x}[\mathbf{y}]$	output

where  $\mathbf{y}$  is a (possibly empty) tuple  $(y_1, \dots, y_n)$  of pairwise distinct names. We denote by  $\mathbf{0}$  the empty sum of processes (that is when  $I = \emptyset$ ). We call a nonempty sum  $\sum_{i \in I} \pi_i.P_i$  of processes an *agent*, and a replication<sup>2</sup>  $!u.P$  a *server*. We use the common abbreviation that consists of omitting the trailing  $\mathbf{0}$  in  $\pi$ -terms. The restriction  $\nu x.P$  binds the name  $x$  in  $P$  and an input guard  $x(y_1, \dots, y_n).P$  binds the names  $y_1, \dots, y_n$  in  $P$ . We denote by  $fn(P)$  the set of names occurring free in a process  $P$ . If  $\mathbf{x}$  and  $\mathbf{y}$  are tuples of names of the same length, we denote by  $\{\mathbf{y}/\mathbf{x}\}$  the substitution that maps any  $x_i$  to  $y_i$ . The standard operational semantics of the  $\pi$ -calculus is given by a structural congruence  $\equiv$  and a reduction relation defined in Fig. 1 and Fig. 2.

<sup>2</sup> We use a restricted form of replication inspired from the one involved in the definition of PICT [Tur95]. Yet it is expressive enough to perform all major constructions in the  $\pi$ -calculus (data structures, higher-order communication, etc.).

Let  $\equiv$  be the smallest congruence relation on  $\mathcal{P}$  which satisfies the following axioms:

- $P \equiv Q$  if  $P$  and  $Q$  are  $\alpha$ -convertible.
- $(\mathcal{P}/\equiv, |, \mathbf{0})$  is a commutative monoid.
- $\nu x.\nu y.P \equiv \nu y.\nu x.P$ .
- $\nu x.\mathbf{0} \equiv \mathbf{0}$ .
- $\nu x.(P | Q) \equiv P | \nu x.Q$  whenever  $x$  is not free in  $P$ .

---

**Fig. 1.** Structural congruence

---

$$\frac{P \rightarrow P'}{P | Q \rightarrow P' | Q} \qquad \frac{P \rightarrow P'}{\nu x.P \rightarrow \nu x.P'}$$

$$(\dots + x(\mathbf{y}).P + \dots) | (\dots + \bar{x}[\mathbf{z}].Q + \dots) \rightarrow \{\mathbf{z}/\mathbf{y}\}P | Q$$

$$!x(\mathbf{y}).P | (\dots + \bar{x}[\mathbf{z}].Q + \dots) \rightarrow \{\mathbf{z}/\mathbf{y}\}P | !x(\mathbf{y}).P | Q$$

$$\frac{P \equiv P' \quad P' \rightarrow Q' \quad Q' \equiv Q}{P \rightarrow Q}$$


---

**Fig. 2.** Process reduction

We model a system  $S$  of mobile processes by a closed  $\pi$ -term. The evolution of the system is determined by the internal communications. The structural laws allow us to rewrite a  $\pi$ -term in order to bring communicands in juxtaposition. They are the analogue of the heating/cooling rules of the Chemical Abstract Machine [BB92]. In the chemical metaphore, a  $\pi$ -term is a syntactic encoding of a “solution of floating ions”, i.e. a multiset of agents and servers, in which reaction is communication. The structural congruence identifies all encodings of the same solution. In particular, any closed process  $P$  is structurally equivalent to a term of the following form:

$$\nu x_1 \dots \nu x_l.(A_1 | \dots | A_m | S_1 | \dots | S_n) \quad (\star)$$

where the  $A_i$  are agents and the  $S_j$  are servers, such that each name occurring free in any one of the  $A_i$  or  $S_j$  lies in  $\{x_1, \dots, x_l\}$ . Intuitively, this  $\pi$ -term exhibits the *communication topology* of  $P$ , i.e. all active agents and servers linked together by communication channels. Whereas the  $A_i$  and  $S_j$  are unique (up to permutation of the indices), the decomposition  $(\star)$  is not, because of  $\alpha$ -renaming and the possibility to add and remove useless channel names by structural rewriting<sup>3</sup>.

<sup>3</sup> If  $x \notin fn(P)$ ,  $\nu x.P \equiv P$ .



where boxes (resp. circles) represent hyperedges (resp. vertices).  $\square$

Our goal is to discover automatically a finitely representable superset of  $\mathcal{T}_c^*(S)$ . We will apply techniques coming from the area of alias analysis [Ven96], but this requires first to refine the standard semantics of the  $\pi$ -calculus.

### 3 Nonstandard Semantics of the $\pi$ -Calculus

Let  $S$  be a closed  $\pi$ -term. A careful inspection of the reduction rules of Fig. 2 shows that any active agent or server of a process  $P$  such that  $S \xrightarrow{*} P$ , is a *residual* of a subprocess of  $S$ , i.e. a process  $sQ$ , where  $s$  is a substitution and  $Q$  is a subterm of  $S$ . In particular, several residuals of a same subprocess of  $S$  can only be obtained by repeated requests to a server via communication. Thereby, we can uniquely identify an active process in  $P$  with the sequence of requests to servers during the computation  $S \xrightarrow{*} P$  that contributed to create the process, together with the subprocess of  $S$  from which it is a residual<sup>4</sup>. We use this idea to construct a new operational semantics of the system  $S$  of mobile processes.

In order to simplify the presentation of the semantics, we assume that  $S$  is in *normal form*, given by the following grammar:

$$\begin{aligned} N &::= \mathbf{0} \mid \nu x_1 \dots \nu x_m.(AS_1 \mid \dots \mid AS_n) \\ AS &::= !\iota.N \mid \sum_{i \in I} \pi_i.N_i \end{aligned}$$

where  $I \neq \emptyset$  and  $\iota, \pi$  are the atomic actions of the  $\pi$ -calculus defined in Sect. 2. Moreover, we require that all variable bindings in  $S$  be pairwise distinct. Any process can always be brought into normal form by using the structural laws. We uniquely label each occurrence of an agent or a server in  $S$ . Let  $\mathcal{L}(S)$  be the set of these labels, and  $\mathcal{A}(S)$  be the subset corresponding to all labels of agents. For any  $\ell \in \mathcal{L}(S)$ , we denote by  $S_\ell$  the process labelled by  $\ell$ . A configuration in our operational semantics is a pair  $(\Phi, \rightsquigarrow)$  where:

1.  $\Phi$  is a function from  $\mathcal{L}(S)$  into  $\wp(\mathcal{A}(S)^*)$  such that, for each  $\ell \in \mathcal{L}(S)$ ,  $\Phi(\ell)$  is finite.
2.  $\rightsquigarrow$  is an equivalence relation over the following set:

$$Ports(\Phi) \stackrel{\text{def}}{=} \{(\ell, L, x) \mid \ell \in \mathcal{L}(S), L \in \Phi(\ell), x \in fn(S_\ell)\}$$

The function  $\Phi$  associates to each label  $\ell$  the set of residuals of  $S_\ell$  in the configuration. A residual is identified with the sequence of agent labels which were involved in the creation of the process via requests to servers.  $Ports(\Phi)$  is the set of communication ports corresponding to the distribution of processes  $\Phi$ . The communication channels are given by the congruence classes of the channel relation  $\rightsquigarrow$  which expresses the links between ports. Note that since we use the  $\rightsquigarrow$  relation, we get rid of problems of name generation and  $\alpha$ -conversion

<sup>4</sup> This idea is somehow related to the notion of *locality* developed for process algebras [BCHK94, San94b].

as it is the case in the standard semantics. The nonstandard semantics gives more information than what is really needed to define the communication topology. However, this higher level of expressivity is required to design an accurate abstract interpretation, as we will see in Sect. 5.

We denote by  $\mathcal{C}(S)$  the set of all configurations  $(\Phi, \rightsquigarrow)$ . If  $\Phi \in \mathcal{L}(S) \longrightarrow \wp(\mathcal{A}(S)^*)$  and  $\rho$  is a binary relation over  $\text{Ports}(\Phi)$ , we denote by  $[\rho]_\Phi$  the smallest equivalence relation over  $\text{Ports}(\Phi)$  containing  $\rho$ . We denote by  $\dot{\cup}$  the pointwise inclusion on  $\mathcal{L}(S) \longrightarrow \wp(\mathcal{A}(S)^*)$ . The nonstandard operational semantics of  $S$  is given by a transition relation  $\Rightarrow$  on  $\mathcal{C}(S)$  defined in Fig. 3 and Fig. 4. The semantic rules require a few explanations. In the case of communication between two agents (Fig. 3), the instances of  $S_{\ell_i}$  and  $S_{\ell_o}$  are removed from  $\Phi$  and new instances of the  $S_{\ell_i^j}$  and  $S_{\ell_o^k}$  are created with the same residual as their parent process. The channel relation is defined on the process distribution  $\Phi \dot{\cup} \Phi'$  including the communicands and the newly released processes. The relation  $\rightsquigarrow_c$  implements communication, i.e. it links  $y_i$  and  $t_i$  in all spawned processes that involve these names. The rôle of  $\rightsquigarrow_i$  and  $\rightsquigarrow_o$  is to bind the free names in the  $S_{\ell_i^j}$  and  $S_{\ell_o^k}$  to the corresponding ones in the parent processes. Finally  $\rightsquigarrow_\nu$  relates all occurrences of a newly created name among the processes spawned by the sender and the receiver. This information is then propagated to all other ports by the transitive closure involved in  $[-]_{\Phi \dot{\cup} \Phi'}$ . Once this has been done, the restriction of  $\rightsquigarrow'$  to  $\text{Ports}(\Phi')$  eliminates all information relative to the communicands. The situation is almost identical for a communication between an agent and a server (Fig. 4), except that the instance of the server is not deleted. The newly spawned instances of the  $S_i^j$  are assigned the residual  $L_o.\ell_o$ , thus formalizing our intuition about the meaning of residuals.

In our context, given a closed  $\pi$ -term  $S$  in normal form, we are interested in computing an upper-approximate of the *collecting semantics* [CC77]  $\mathcal{S}_S$  of  $S$ , that is the set  $\{P \mid S \xrightarrow{*} P\}$  of descendants of the initial configuration in the standard operational semantics. Thereby, we connect the nonstandard operational semantics with the standard one via a relation  $\kappa \in \wp(\mathcal{P} \times \mathcal{C}(S))$  defined as follows. If  $\mathcal{L}(S) = \{\ell_1, \dots, \ell_n\}$  and, for all  $i \in \{1, \dots, n\}$ ,  $\Phi(\ell_i) = \{L_{\ell_i,1}, \dots, L_{\ell_i,k_i}\}$ , then

$$(p, (\Phi, \rightsquigarrow)) \in \kappa$$

whenever  $p \equiv \nu x_1 \dots \nu x_m.(P_{\ell_1,1} \mid \dots \mid P_{\ell_1,k_1} \mid \dots \mid P_{\ell_n,1} \mid \dots \mid P_{\ell_n,k_n})$  where:

- For all  $i \in \{1, \dots, n\}$  and  $j \in \{1, \dots, k_i\}$ ,  $P_{\ell_i,j} = s_{i,j}S_{\ell_i}$ , where  $s_{i,j}$  is a substitution such that  $s_{i,j}(\text{fn}(S_{\ell_i})) \subseteq \{x_1, \dots, x_m\}$ .
- $\forall i, j \in \{1, \dots, n\} : \forall x \in \text{fn}(S_{\ell_i}) : \forall y \in \text{fn}(S_{\ell_j}) : \forall p \in \{1, \dots, k_i\} : \forall q \in \{1, \dots, k_j\} : s_{\ell_i,p}(x) = s_{\ell_j,q}(y) \iff (\ell_i, L_{\ell_i,p}, x) \rightsquigarrow (\ell_j, L_{\ell_j,q}, y)$ .

If  $S = \nu x_1 \dots \nu x_l.(S_{\ell_1} \mid \dots \mid S_{\ell_n})$  we denote by  $c_S$  the nonstandard configuration  $(\Phi_S, \rightsquigarrow_S)$  where

$$\Phi_S(\ell) = \begin{cases} \{\varepsilon\} & \text{if } \ell = \ell_i \text{ for } i \in \{1, \dots, n\} \\ \emptyset & \text{otherwise} \end{cases}$$

and  $\rightsquigarrow_S = [\{((\ell_i, \varepsilon, x), (\ell_j, \varepsilon, x)) \mid 1 \leq i, j \leq n, x \in \text{fn}(S_{\ell_i}) \cap \text{fn}(S_{\ell_j})\}]_{\Phi_S}$ . Note that  $(S, c_S) \in \kappa$ . The nonstandard collecting semantics  $\mathcal{S}_S^{\natural}$  is given by  $\{c \mid c_S \xrightarrow{*} c\}$ .

**Theorem 2.** *Let  $\Gamma : \wp(\mathcal{C}(S)) \longrightarrow \wp(\mathcal{P})$  be the function that sends any set  $C$  of configurations to  $\{p \mid \exists c \in C : (p, c) \in \kappa\}$ . Then,  $\mathcal{S}_S = \Gamma(\mathcal{S}_S^{\natural})$ .*

This means that the standard and nonstandard collecting semantics of the  $\pi$ -calculus are equivalent. Note that  $\kappa$  provides a direct way to compute the communication topology associated to a nonstandard configuration  $(\Phi, \rightsquigarrow)$ , the vertices being given by the congruence classes of  $\rightsquigarrow$ .

*Example 2.* Consider the  $\pi$ -term  $S$  of Example 1 (which is in normal form) that we label as follows:

$$\nu l. \nu c. \nu x. (\boxed{\ell_1 : \bar{l}[x]} \mid \boxed{\ell_2 : !l(y). \nu z. (\boxed{\ell_3 : \bar{c}[y, z]} \mid \boxed{\ell_4 : \bar{l}[z]})})$$

If  $c_S \xrightarrow{n} (\Phi, \rightsquigarrow)$ , for  $n > 0$ , then

$$\Phi = \left\{ \begin{array}{l} \ell_1 \mapsto \emptyset \\ \ell_2 \mapsto \{\varepsilon\} \\ \ell_3 \mapsto \{\ell_1. \ell_4^i \mid 0 \leq i < n\} \\ \ell_4 \mapsto \{\ell_1. \ell_4^{n-1}\} \end{array} \right\}$$

and  $\rightsquigarrow$  is the smallest equivalence relation on  $\text{Ports}(\Phi)$  such that:

- $(\ell_3, \ell_1. \ell_4^i, y) \rightsquigarrow (\ell_3, \ell_1. \ell_4^j, z) \iff i = j + 1$
- $(\ell_4, \ell_1. \ell_4^{n-1}, z) \rightsquigarrow (\ell_3, \ell_1. \ell_4^{n-1}, z)$
- $(\ell_4, \ell_1. \ell_4^{n-1}, l) \rightsquigarrow (\ell_2, \varepsilon, l)$
- $(\ell_3, \ell_1. \ell_4^i, c) \rightsquigarrow (\ell_2, \varepsilon, c)$ , for all  $i \in \{0, \dots, n-1\}$  □

It now remains to build an abstract interpretation of the  $\pi$ -calculus based upon the nonstandard semantics.

## 4 Abstract Interpretation

Abstract Interpretation [CC77, CC92] provides general frameworks to reason about *semantic approximation*. Typically, this problem amounts to finding an upper-approximate of the least fixpoint  $\mathcal{S}^{\natural}$  of a  $\sqcup^{\natural}$ -complete endomorphism  $F^{\natural}$  over a complete lattice  $(\mathcal{D}^{\natural}, \sqsubseteq^{\natural}, \perp^{\natural}, \sqcup^{\natural}, \top^{\natural}, \cap^{\natural})$ , the *concrete semantic function*.

*Example 3.* The nonstandard collecting semantics  $\mathcal{S}_S^{\natural}$  of a system  $S$  of mobile processes can be shown to be equal to the least fixpoint of the  $\sqcup$ -complete morphism  $F_S$  defined over  $(\wp(\mathcal{C}(S)), \subseteq, \emptyset, \cup, \mathcal{C}(S), \cap)$  as follows (see [CC77]):

$$F_S \stackrel{\text{def}}{=} \lambda X \cdot \{c_S\} \cup \{c \mid \exists c' \in X : c' \Rightarrow c\}$$

We can thereby apply an abstract interpretation framework to the analysis of the  $\pi$ -calculus. □

---

If

- $\ell_i, \ell_o \in \mathcal{L}(S)$ ,
- $L_i \in \Phi(\ell_i), L_o \in \Phi(\ell_o)$ ,
- $S_{\ell_i} = \dots + x(y_1, \dots, y_m) \cdot \nu v_1^i \dots \nu v_{i_i}^i \cdot (S_{\ell_i^1} \mid \dots \mid S_{\ell_i^{n_i}}) + \dots$ ,
- $S_{\ell_o} = \dots + \bar{z}[t_1, \dots, t_m] \cdot \nu v_1^o \dots \nu v_{i_o}^o \cdot (S_{\ell_o^1} \mid \dots \mid S_{\ell_o^{n_o}}) + \dots$ ,
- $(\ell_i, L_i, x) \rightsquigarrow (\ell_o, L_o, z)$ ,

then

$$(\Phi, \rightsquigarrow) \Rightarrow (\Phi', \rightsquigarrow')$$

where:

- If we put

$$\Phi^\dagger(\ell) = \begin{cases} \Phi(\ell) - \{L_\eta\} & \text{if } \ell = \ell_\eta \text{ for } \eta \in \{i, o\} \\ \Phi(\ell) & \text{otherwise} \end{cases}$$

then

$$\Phi'(\ell) = \begin{cases} \Phi^\dagger(\ell) \cup \{L_\eta\} & \text{if } \ell = \ell_\eta^k \text{ for } \eta \in \{i, o\}, k \in \{1, \dots, n_\eta\} \\ \Phi^\dagger(\ell) & \text{otherwise} \end{cases}$$

- $\rightsquigarrow' = [\rightsquigarrow \cup \rightsquigarrow_c \cup \rightsquigarrow_i \cup \rightsquigarrow_o \cup \rightsquigarrow_\nu]_{\Phi \cup \Phi'} \cap (Ports(\Phi') \times Ports(\Phi'))$  where:
    - $\rightsquigarrow_c = \{((\ell_i^j, L_i, y_h), (\ell_o^k, L_o, t_h)) \mid 1 \leq j \leq n_i, 1 \leq k \leq n_o, 1 \leq h \leq m, y_h \in fn(S_{\ell_i^j}), t_h \in fn(S_{\ell_o^k})\}$ .
    - $\rightsquigarrow_\eta = \{((\ell_\eta, L_\eta, u), (\ell_\eta^j, L_\eta, u)) \mid 1 \leq j \leq n_\eta, u \in fn(S_{\ell_\eta}) \cap fn(S_{\ell_\eta^j})\}$ , for  $\eta \in \{i, o\}$ .
    - $\rightsquigarrow_\nu = \{((\ell_\eta^j, L_\eta, v_h^\eta), (\ell_\eta^k, L_\eta, v_h^\eta)) \mid j, k \in \{1, \dots, n_\eta\}, h \in \{1, \dots, l_\eta\}, v_h^\eta \in fn(S_{\ell_\eta^j}) \cap fn(S_{\ell_\eta^k})\}$ , for  $\eta \in \{i, o\}$ .
- 

**Fig. 3.** Interagent communication

Following [CC92], the semantic approximation is achieved via an *abstract semantic specification* given by a preordered set  $(\mathcal{D}^\sharp, \preceq)$ , the *abstract semantic domain*, related to  $\mathcal{D}^\natural$  by a *concretization function*  $\gamma : \mathcal{D}^\sharp \longrightarrow \mathcal{D}^\natural$ , an *abstract basis*  $\perp^\sharp \in \mathcal{D}^\sharp$ , and an *abstract semantic function*  $F^\sharp : \mathcal{D}^\sharp \longrightarrow \mathcal{D}^\sharp$ , such that:

1.  $\perp^\natural \sqsubseteq^\natural \gamma(\perp^\sharp)$ .
2.  $\forall x, y \in \mathcal{D}^\sharp : x \preceq y \implies \gamma(x) \sqsubseteq^\natural \gamma(y)$ .
3.  $\forall x \in \mathcal{D}^\sharp : F^\natural \circ \gamma(x) \sqsubseteq^\natural \gamma \circ F^\sharp(x)$ .

In order to compute effectively an approximation of  $\mathcal{S}^\natural$ , we introduce the notion of *widening operator*.

**Definition 3 Widening Operator [CC77, CC92].** A *widening* on  $(\mathcal{D}^\sharp, \preceq)$  is a binary operator  $\nabla : \mathcal{D}^\sharp \times \mathcal{D}^\sharp \longrightarrow \mathcal{D}^\sharp$  which satisfies the following properties:

1.  $\forall x, y \in \mathcal{D}^\sharp : x \preceq x \nabla y$ .

---

If

- $\ell_i, \ell_o \in \mathcal{L}(S)$ ,
- $L_i \in \Phi(\ell_i), L_o \in \Phi(\ell_o)$ ,
- $S_{\ell_i} = !x(y_1, \dots, y_m). \nu v_1^i \dots \nu v_{l_i}^i. (S_{\ell_i^1} \mid \dots \mid S_{\ell_i^{n_i}})$ ,
- $S_{\ell_o} = \dots + \bar{z}[t_1, \dots, t_m]. \nu v_1^o \dots \nu v_{l_o}^o. (S_{\ell_o^1} \mid \dots \mid S_{\ell_o^{n_o}}) + \dots$ ,
- $(\ell_i, L_i, x) \rightsquigarrow (\ell_o, L_o, z)$ ,

then

$$(\Phi, \rightsquigarrow) \Rightarrow (\Phi', \rightsquigarrow')$$

where:

- If we put

$$\Phi^\dagger(\ell) = \begin{cases} \Phi(\ell) - \{L_o\} & \text{if } \ell = \ell_o \\ \Phi(\ell) & \text{otherwise} \end{cases}$$

then

$$\Phi'(\ell) = \begin{cases} \Phi^\dagger(\ell) \cup \{L_o\} & \text{if } \ell = \ell_o^k \text{ for } k \in \{1, \dots, n_o\} \\ \Phi^\dagger(\ell) \cup \{L_o.\ell_o\} & \text{if } \ell = \ell_i^k \text{ for } k \in \{1, \dots, n_i\} \\ \Phi^\dagger(\ell) & \text{otherwise} \end{cases}$$

- $\rightsquigarrow' = [\rightsquigarrow \cup \rightsquigarrow_c \cup \rightsquigarrow_i \cup \rightsquigarrow_o \cup \rightsquigarrow_\nu]_{\Phi \cup \Phi'} \cap (Ports(\Phi') \times Ports(\Phi'))$  where:
    - $\rightsquigarrow_c = \{((\ell_i^j, L_o.\ell_o, y_h), (\ell_o^k, L_o, t_h)) \mid 1 \leq j \leq n_i, 1 \leq k \leq n_o, 1 \leq h \leq m, y_h \in fn(S_{\ell_i^j}), t_h \in fn(S_{\ell_o^k})\}$ .
    - $\rightsquigarrow_i = \{((\ell_i, L_i, u), (\ell_i^j, L_o.\ell_o, u)) \mid 1 \leq j \leq n_i, u \in fn(S_{\ell_i}) \cap fn(S_{\ell_i^j})\}$ .
    - $\rightsquigarrow_o = \{((\ell_o, L_o, u), (\ell_o^j, L_o, u)) \mid 1 \leq j \leq n_o, u \in fn(S_{\ell_o}) \cap fn(S_{\ell_o^j})\}$ .
    - $\rightsquigarrow_\nu = \{((\ell_o^j, L_o, v_h^o), (\ell_o^k, L_o, v_h^o)) \mid j, k \in \{1, \dots, n_o\}, h \in \{1, \dots, l_o\}, v_h^o \in fn(S_{\ell_o^j}) \cap fn(S_{\ell_o^k})\} \cup \{((\ell_i^j, L_o.\ell_o, v_h^i), (\ell_i^k, L_o.\ell_o, v_h^i)) \mid j, k \in \{1, \dots, n_i\}, h \in \{1, \dots, l_i\}, v_h^i \in fn(S_{\ell_i^j}) \cap fn(S_{\ell_i^k})\}$ .
- 

**Fig. 4.** Agent-server communication

2.  $\forall x, y \in \mathcal{D}^\# : y \preceq x \nabla y$ .
3. For every sequence  $(x_n)_{n \geq 0}$  of elements of  $\mathcal{D}^\#$ , the sequence  $(x_n^\nabla)_{n \geq 0}$  inductively defined as follows:

$$\begin{cases} x_0^\nabla & = x_0 \\ x_{n+1}^\nabla & = x_n^\nabla \nabla x_{n+1} \end{cases}$$

is ultimately stationary. □

The approximation of the least fixpoint is obtained as the limit of an *abstract iteration sequence* with widening.

**Theorem 4 Abstract Iterates [CC92].** *The iteration sequence  $(F_n^\nabla)_{n \geq 0}$  defined as:*

$$\begin{cases} F_0^\nabla &= \perp^\sharp \\ F_{n+1}^\nabla &= F_n^\nabla & \text{if } F(F_n^\nabla) \preceq F_n^\nabla \\ &= F_n^\nabla \nabla F(F_n^\nabla) & \text{otherwise} \end{cases}$$

*is ultimately stationary and its limit  $S^\sharp$  satisfies  $S^\sharp \sqsubseteq^\sharp \gamma(S^\sharp)$ . Moreover, if  $N \geq 0$  is such that  $F_N^\nabla = F_{N+1}^\nabla$ , then  $\forall n \geq N : F_n^\nabla = F_N^\nabla$ .*

We now have to construct an abstract domain  $(\mathcal{D}^\sharp, \preceq)$  approximating  $\mathcal{C}(S)$  via a concretization function  $\gamma : \mathcal{D}^\sharp \longrightarrow \mathcal{C}(S)$ .

## 5 Abstract Semantic Domain

We construct  $\mathcal{D}^\sharp$  as a set of abstract configurations  $(\Phi^\sharp, \rightsquigarrow^\sharp)$  defined as follows. We represent a set of residuals of a process  $S_\ell$  by a rational language. More precisely, let  $\mathbb{A}$  be the set of automata  $(Q, I, T, \tau)$ , where  $Q$  is a finite set of states,  $I, T \subseteq Q$  are the initial and final states, and  $\tau \in \wp(Q \times \mathcal{A}(S) \times Q)$  is the transition relation. In order to keep  $\mathbb{A}$  small and representable, we suppose that all states come from an infinite recursive set  $\mathcal{Q}$ . For any automaton  $\mathcal{A}$  in  $\mathbb{A}$ , we denote by  $Paths(\mathcal{A})$  the set of successful paths of  $\mathcal{A}$ , and by  $\tau(\mathcal{A})$  the set of transitions of  $\mathcal{A}$ . An abstract distribution of processes  $\Phi^\sharp$  is a function from  $\mathcal{L}(S)$  into  $\mathbb{A}$ . An abstract communication port is a tuple  $(\ell, i, t, x)$  where  $\ell \in \mathcal{L}(S)$ ,  $x \in fn(S_\ell)$ ,  $i$  and  $t$  being respectively initial and final states of the automaton  $\Phi^\sharp(\ell)$ . It describes all concrete ports  $(\ell, L, x)$  where  $L$  is a word labelling a path<sup>5</sup> from  $i$  to  $t$  in  $\Phi^\sharp(\ell)$ . We denote by  $Ports^\sharp(\Phi^\sharp)$  the set of all abstract communication ports associated to  $\Phi^\sharp$ .

Our representation of the abstract channel relation is based upon a numerical encoding of paths. The aim of this construction is to be able to represent *non-uniform channel relations*, i.e. to distinguish instances of a process which is recursively spawned. Intuitively, if  $\ell$  is the label of such a process, the automaton  $\Phi^\sharp(\ell)$ , which is an approximation of the residuals of  $S_\ell$ , certainly contains cycles (since there is potentially an infinity of instances of  $S_\ell$ ). The encoding amounts to assigning a counter to each of these cycles and to identify a residual  $L$  of  $S_\ell$  by the number of times a path labelled by  $L$  runs through each cycle, i.e. by a tuple of counter values. This idea was first introduced by A. Deutsch [Deu92a, Deu92b] in the context of alias analysis, with a different formulation though. In order to avoid having to find cycles and also to ensure a better precision, we will actually assign a counter to each arrow of the automaton. We now give a formal definition of this construction.

If  $p$  is a successful path in  $(Q, I, T, \tau)$ , we denote by  $p^\circ$  its *commutative image*, that is the function from  $\tau$  into the set of integers  $\mathbb{N}$ , which maps any transition of the automaton to the number of times it occurs in  $p$ . If  $\ell_1, \ell_2 \in \mathcal{L}(S)$  and

<sup>5</sup> We can reduce the number of abstract ports to consider by requiring that there be at least one path in  $\Phi^\sharp(\ell)$  from  $i$  to  $t$ .

$\Phi^\sharp(\ell_i) = (Q_i, I_i, T_i, \tau_i)$  for  $i \in \{1, 2\}$ , the channel relation between abstract ports is given by a set of tuples  $(\ell_1, i_1, t_1, x) \overset{\nu}{\rightsquigarrow}^\sharp (\ell_2, i_2, t_2, y)$  where  $\nu \in (\tau_1 \longrightarrow \mathbb{N}) \times (\tau_2 \longrightarrow \mathbb{N})$ . If we denote by  $i(p)$  (resp.  $t(p)$ ) the initial (resp. final) state of a successful path  $p$  of an automaton, this abstract channel relation denotes all pairs  $(\ell_1, L_1, x) \rightsquigarrow (\ell_2, L_2, y)$  for which there exists a successful path  $p_j$  in  $\Phi^\sharp(\ell_j)$  labelled by  $L_j$ , such that  $i(p_j) = i_j$  and  $t(p_j) = t_j$ , for each  $j \in \{1, 2\}$ , and  $\nu = (p_1^\circ, p_2^\circ)$ . Considering commutative images of paths instead of tuples of counter values alleviates us from introducing explicit counter variables which would have led to intricate problems of renaming when it had come to defining the abstract semantics. In the following we will tacitly use the isomorphism:

$$(\tau_1 \longrightarrow \mathbb{N}) \times (\tau_2 \longrightarrow \mathbb{N}) \cong (\tau_1 \oplus \tau_2) \longrightarrow \mathbb{N}$$

where  $\oplus$  denotes the coproduct in the category **Sets** of sets and functions (i.e. the disjoint union).

In order to guarantee finite representability for the abstract configurations, we require that the set:

$$(\ell_1, i_1, t_1, x) \rightsquigarrow^\sharp (\ell_2, i_2, t_2, y) \stackrel{\text{def}}{=} \{\nu \mid (\ell_1, i_1, t_1, x) \overset{\nu}{\rightsquigarrow}^\sharp (\ell_2, i_2, t_2, y)\}$$

be effectively given. This can be achieved by considering an *abstract numerical domain*, that is a representable class of subsets of  $(\tau_1 \oplus \tau_2) \longrightarrow \mathbb{N}$ . Several numerical domains have already been developed for abstract interpretation using various kinds of representations: *intervals* [CC76], *arithmetic congruences* [Gra89], *affine equalities* [Kar76], *convex polyhedra* [CH78] or *congruence equalities* [Gra91]. They all can be described in an abstract way. Let **FinSets** be the category of finite sets and functions. If  $f : V \longrightarrow W$  is a function between finite sets, we define  $\mathcal{V}f : \wp(V \longrightarrow \mathbb{N}) \longrightarrow \wp(W \longrightarrow \mathbb{N})$  to be the function that maps any  $X \in \wp(V \longrightarrow \mathbb{N})$  to the set:

$$\{\mu : W \longrightarrow \mathbb{N} \mid \exists \nu \in X : \mu = \lambda w \cdot \sum_{v \in f^{-1}(w)} \nu(v)\}$$

**Definition 5 Abstract Numerical Domain.** An abstract numerical domain is a functor  $\mathcal{V}^\sharp : \mathbf{FinSets} \longrightarrow \mathbf{PoSets}$  that sends any finite set  $V$  to a poset that comes with the structure of a lattice  $(\mathcal{V}^\sharp V, \subseteq^\sharp, \sqcup_V^\sharp, \emptyset_V^\sharp, \sqcap_V^\sharp, \top_V^\sharp)$  and a monotone map  $\gamma_V : (\mathcal{V}^\sharp V, \subseteq^\sharp) \longrightarrow (\wp(V \longrightarrow \mathbb{N}), \subseteq)$ . We will omit the subscript  $V$  whenever it will be clear from the context. Moreover, if  $V \longrightarrow W$  is an arrow in **FinSets**, then  $\mathcal{V}f \circ \gamma_V \subseteq \gamma_W \circ \mathcal{V}^\sharp f$ .  $\square$

*Example 4.* We denote by  $\mathbb{Q}$  the field of rational numbers. Karr's abstract numerical domain [Kar76] associates to each finite set  $V$  the set of all affine subspaces of  $V \longrightarrow \mathbb{Q}$  endowed with its canonical affine structure<sup>6</sup>.  $\gamma_V$  sends any element

<sup>6</sup> An affine subspace of  $V \longrightarrow \mathbb{Q}$  is a set  $\{p + \lambda_1.e_1 + \dots + \lambda_n.e_n \mid \lambda_1, \dots, \lambda_n \in \mathbb{Q}\}$  where  $p, e_1, \dots, e_n$  are elements of  $V \longrightarrow \mathbb{Q}$ , addition and multiplication by a scalar being defined componentwise.

$X^\#$  of  $\mathcal{V}^\#V$  to  $X^\# \cap (V \longrightarrow \mathbb{N})$ , whereas  $\sqcap_V^\#$  is the intersection and  $\sqcup_V^\#$  the sum of affine subspaces. An affine subspace of  $V \longrightarrow \mathbb{Q}$  can equivalently be seen as the set of solutions  $\mathbf{Sol}(\mathbf{S})$  of a system  $\mathbf{S}$  of affine equations  $a_1.v_1 + \dots + a_n.v_n = c$  over the variables  $V = \{v_1, \dots, v_n\}$ .  $\square$

Thereby, an abstract channel relation  $\rightsquigarrow^\#$  can equivalently be seen as a function of domain  $Ports^\#(\Phi^\#) \times Ports^\#(\Phi^\#)$  that maps any pair  $((\ell_1, i_1, t_1, x), (\ell_2, i_2, t_2, y))$  of abstract ports to the element  $(\ell_1, i_1, t_1, x) \rightsquigarrow^\# (\ell_2, i_2, t_2, y)$  of  $\mathcal{V}^\#(\tau_1 \oplus \tau_2)$ . We leave the choice of the abstract numerical domain  $\mathcal{V}^\#$  as a parameter of our analysis.

Following [Ven96], we provide  $\mathcal{D}^\#$  with the structure of a *cofibered domain*. Given two automata  $\mathcal{A}_1 = (Q_1, I_1, T_1, \tau_1)$  and  $\mathcal{A}_2 = (Q_2, I_2, T_2, \tau_2)$ , we define a morphism  $\mathcal{A}_1 \xrightarrow{f} \mathcal{A}_2$  as a pair of functions  $f_1 : Q_1 \longrightarrow Q_2$  and  $f_2 : \tau_1 \longrightarrow \tau_2$  satisfying the following conditions:

1.  $f_1(I_1) \subseteq I_2$ .
2.  $f_1(T_1) \subseteq T_2$ .
3.  $\forall (q, \ell, q') \in \tau_1 : f_2(q, \ell, q') = (f_1(q), \ell, f_1(q'))$ .

$\mathbb{A}$  can thereby be turned into a category. We denote by  $\Phi^\#(S)$  the product category  $\prod_{\ell \in \mathcal{L}(S)} \mathbb{A}$ . An object of  $\Phi^\#(S)$  is an abstract distribution of processes previously defined. We now construct a functor  $\Delta$  from  $\Phi^\#(S)$  into the category **PoSets** of partially ordered sets and monotone maps. The image of an object  $\Phi^\#$  in  $\Phi^\#(S)$  is the set of all abstract channel relations  $\rightsquigarrow^\#$  over  $\Phi^\#$  ordered by inclusion. We call  $\Delta\Phi^\#$  the *fiber* of  $\Delta$  over  $\Phi^\#$ . Now let  $\Phi_1^\# \xrightarrow{\mathcal{F}} \Phi_2^\#$  be a morphism<sup>7</sup> in  $\Phi^\#(S)$ .  $\Delta\mathcal{F}$  maps any abstract channel relation  $\rightsquigarrow^\#_1$  in  $\Phi_1^\#$  to the relation  $\rightsquigarrow^\#_2$  such that, for all  $(\ell_1, i'_1, t'_1, x), (\ell_2, i'_2, t'_2, y) \in Ports^\#(\Phi_2^\#)$ , the set  $(\ell_1, i'_1, t'_1, x) \rightsquigarrow^\#_2 (\ell_2, i'_2, t'_2, y)$  is defined as:

$$\sqcup_{\tau_1 \oplus \tau_2}^\# \{ \mathcal{V}^\#(\mathcal{F}(\ell_1)_2 \oplus \mathcal{F}(\ell_2)_2)((\ell_1, i_1, t_1, x) \rightsquigarrow^\#_1 (\ell_2, i_2, t_2, y)) \mid \begin{array}{l} \mathcal{F}(\ell_j)_1(i_j) = i'_j, \\ \mathcal{F}(\ell_j)_1(t_j) = t'_j, \\ 1 \leq j \leq 2 \end{array} \}$$

where  $\Phi_2^\#(\ell_j) = (Q_j, I_j, T_j, \tau_j)$ , for  $j \in \{1, 2\}$ . We define the preorder  $\preceq$  on  $\mathcal{D}^\#$  from the *Grothendieck construction* [BW90] applied to  $\Delta$ , i.e.  $(\Phi_1^\#, \rightsquigarrow^\#_1) \preceq (\Phi_2^\#, \rightsquigarrow^\#_2)$  iff there exists a morphism  $\Phi_1^\# \xrightarrow{\mathcal{F}} \Phi_2^\#$  such that  $\Delta\mathcal{F}(\rightsquigarrow^\#_1) \subseteq \rightsquigarrow^\#_2$ . We call  $\Delta$  the *display* associated to the cofibered domain  $(\mathcal{D}^\#, \preceq)$ . Intuitively,  $\mathcal{D}^\#$  consists of a category of posets “glued” together. The cofibered structure will play a major rôle in the definition of the abstract semantics and widening operators.

The concretization function  $\gamma : \mathcal{D}^\# \longrightarrow \mathcal{C}(S)$  sends any abstract configuration  $(\Phi^\#, \rightsquigarrow^\#)$  to the set of concrete configurations  $(\Phi, \rightsquigarrow)$  for which there exists a family of functions  $(\Lambda_\ell : \Phi(\ell) \longrightarrow Paths(\Phi^\#(\ell)))_{\ell \in \mathcal{L}(S)}$  satisfying the following conditions:

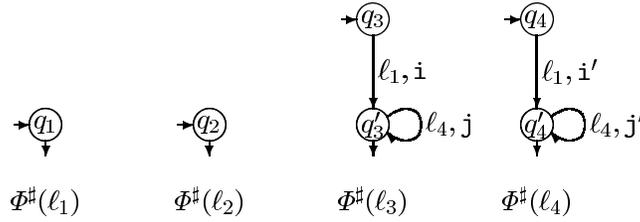
<sup>7</sup> For each  $\ell \in \mathcal{L}(S)$ ,  $\mathcal{F}(\ell) : \Phi_1^\#(\ell) \longrightarrow \Phi_2^\#(\ell)$  is a morphism of automata.

1. For all  $\ell \in \mathcal{L}(S)$  and  $L \in \Phi(\ell)$ , the path  $A_\ell(L)$  is labelled by  $L$ .
2. Let  $(\ell_1, L_1, x), (\ell_2, L_2, y)$  be distinct<sup>8</sup> ports of  $\Phi$ , and  $p_j$  be the path  $A_{\ell_j}(L_j)$ , for  $j \in \{1, 2\}$ . We denote by  $\tau_j, j \in \{1, 2\}$ , the set of transitions of  $\Phi^\#(\ell_j)$ . Then,  $(\ell_1, L_1, x) \rightsquigarrow^\# (\ell_2, L_2, y)$  implies that

$$(p_1^\circ, p_2^\circ) \in \gamma_{\tau_1 \oplus \tau_2}((\ell_1, i(p_1), t(p_1), x) \rightsquigarrow^\# (\ell_2, i(p_2), t(p_2), y))$$

**Proposition 6.**  $\forall c_1^\#, c_2^\# \in \mathcal{D}^\# : c_1^\# \preceq c_2^\# \implies \gamma(c_1^\#) \subseteq \gamma(c_2^\#)$ .

*Example 5.* We carry on with Example 2. Let  $\Phi^\# \in \Phi^\#(S)$  be the abstract distribution of processes defined by the following automata:



where we associate distinct counter variables  $\{i, j, i', j'\}$  to each transition of the automata. If  $\ell_1, \ell_2 \in \mathcal{L}(S)$ , and  $c$  is a counter variable associated to a transition  $\sigma \in \tau(\Phi^\#(\ell_i))$ , for  $i \in \{1, 2\}$ , we denote by  $c_i$  a counter variable associated to the canonical image of  $\sigma$  into  $\tau(\Phi^\#(\ell_1)) \oplus \tau(\Phi^\#(\ell_2))$ . We use Karr's abstract numerical domain, hence systems of affine equations represent the abstract values. Now let  $\rightsquigarrow^\#$  be the abstract channel relation of  $\Delta\Phi^\#$  that satisfies:

$$\left\{ \begin{array}{l} (\ell_1, q_1, q_1, l) \rightsquigarrow^\# (\ell_2, q_2, q_2, l) = \top^\# \\ (\ell_1, q_1, q_1, l) \rightsquigarrow^\# (\ell_4, q_4, q_4', l) = \top^\# \\ (\ell_2, q_2, q_2, l) \rightsquigarrow^\# (\ell_4, q_4, q_4', l) = \top^\# \\ (\ell_3, q_3, q_3', c) \rightsquigarrow^\# (\ell_3, q_3, q_3', c) = \top^\# \\ (\ell_3, q_3, q_3', c) \rightsquigarrow^\# (\ell_2, q_2, q_2', c) = \top^\# \\ (\ell_3, q_3, q_3', y) \rightsquigarrow^\# (\ell_3, q_3, q_3', z) = \left\{ \begin{array}{l} i_1 = i_2 = 1 \\ j_1 = j_2 + 1 \end{array} \right\} \\ (\ell_4, q_4, q_4', z) \rightsquigarrow^\# (\ell_3, q_3, q_3', z) = \left\{ \begin{array}{l} i'_1 = i_2 \\ j'_1 = j_2 \end{array} \right\} \end{array} \right.$$

the other components being defined symmetrically. Thus  $\rightsquigarrow^\#$  denotes *exactly* all communication channels that may be created during the evolution of  $S$ .  $\square$

<sup>8</sup> We do not require the channel relation to denote reflexivity since this information can be extracted from  $\Phi$ .

## 6 Abstract Semantics of the $\pi$ -Calculus

Before defining the abstract semantics, we need some additional properties of abstract numerical domains. Let  $V \xrightarrow{f} W$  be an injective map in **FinSets**. Any abstract numerical domain  $\mathcal{V}^\# : \mathbf{FinSets} \rightarrow \mathbf{PoSets}$  provides two computable functions, an *embedding*  $\vec{f} : \mathcal{V}^\#V \rightarrow \mathcal{V}^\#W$ , and a *projection*  $\overleftarrow{f} : \mathcal{V}^\#W \rightarrow \mathcal{V}^\#V$ , satisfying the following conditions:

1.  $\forall X^\# \in \mathcal{V}^\#V : \forall \nu \in \gamma_V(X^\#) : \{\mu \in W \rightarrow \mathbb{N} \mid \mu \circ f = \nu\} \subseteq \gamma_W(\vec{f}(X^\#))$ .
2.  $\forall Y^\# \in \mathcal{V}^\#W : \forall \nu \in \gamma_W(Y^\#) : \nu \circ f \in \gamma_V(\overleftarrow{f}(Y^\#))$ .

If  $\mathbf{S}$  is a system of linear equations over the set of variables  $V$ , there is a computable element  $\mathbf{Sol}_V^\#(\mathbf{S})$  of  $\mathcal{V}^\#V$  which upper-approximates the set of variable assignments in  $V \rightarrow \mathbb{N}$  that are solutions of  $\mathbf{S}$ . We will omit the subscript  $V$  whenever it will be clear from the context. Moreover, the lattice  $\mathcal{V}^\#V$  is provided with a widening operator.

*Example 6.* For Karr's abstract numerical domain,  $\overleftarrow{f}(Y^\#)$  amounts to the elimination of the coordinates corresponding to the variables  $W - f(V)$  from a system of a point and vectors describing  $Y^\#$ .  $\vec{f}(X^\#)$  is described by the same system of affine equations as  $X^\#$ .  $\mathbf{Sol}_V^\#(S)$  is of course directly expressible in  $\mathcal{V}^\#V$ . Since  $\mathcal{V}^\#V$  satisfies the ascending chain condition, a widening is provided by the join  $\sqcup_V^\#$ .  $\square$

We first construct the abstract counterpart of the closure operator  $[-]_\Phi$  that maps any binary relation  $\rho$  over  $Ports(\Phi)$  to the smallest equivalence relation containing  $\rho$ . This closure operation can be expressed by the following inductive rules:

$$\frac{x \in Ports(\Phi)}{(x, x) \in \rho} \qquad \frac{(x, y) \in \rho}{(y, x) \in \rho} \qquad \frac{(x, y) \in \rho \quad (y, z) \in \rho}{(x, z) \in \rho}$$

The computation of  $[\rho]_\Phi$  amounts to calculating a least fixpoint in  $(\wp(Ports(\Phi)) \times Ports(\Phi), \subseteq)$  (see [CC95]). Thereby, we define the corresponding abstract operation as the limit of an abstract iteration sequence<sup>9</sup> on a fiber of  $\mathcal{D}^\#$  using Theorem 4. For this purpose, we introduce a closure relation  $\triangleright$  on abstract channel relations which mimics the inductive rules for  $[-]_\Phi$ . Let  $\Phi^\#$  be an abstract distribution of processes. We define  $\triangleright$  on  $\Delta\Phi^\#$  as follows:

1. If  $pt'_1, pt'_2 \in Ports^\#(\Phi^\#)$  and  $\rightsquigarrow^\#_1 \in \Delta\Phi^\#$ , then  $\rightsquigarrow^\#_1 \triangleright \rightsquigarrow^\#_2$  where, for all  $pt_1, pt_2 \in Ports^\#(\Phi^\#)$ :

$$pt_1 \rightsquigarrow^\#_2 pt_2 = \begin{cases} pt_1 \rightsquigarrow^\#_1 pt_2 & \text{if } pt_1 = pt'_1 \text{ and } pt_2 = pt'_2 \\ \emptyset^\# & \text{otherwise} \end{cases}$$

<sup>9</sup> Note that we only have to cope with the symmetry and transitivity rules (cf. the definition of  $\gamma$ ).

2. Let  $\ell_1, \ell_2, \ell_3 \in \mathcal{L}(S)$  and  $pt'_j = (\ell_j, i_j, t_j, x_j) \in Ports^\sharp(\Phi^\sharp)$ , for  $j \in \{1, 2, 3\}$ . We denote by  $(Q_j, I_j, T_j, \tau_j)$  the automaton  $\Phi^\sharp(\ell_j)$ , for  $j \in \{1, 2, 3\}$ . Let  $\tau = \tau_1 \oplus \tau_2 \oplus \tau_3$  and  $\tau_1 \oplus \tau_2 \xrightarrow{f_1} \tau$ ,  $\tau_2 \oplus \tau_3 \xrightarrow{f_2} \tau$ ,  $\tau_1 \oplus \tau_3 \xrightarrow{f_3} \tau$  be the canonical inclusion maps in **FinSets**. Now let  $\rightsquigarrow^\sharp_1 \in \Delta\Phi^\sharp$  and  $X_j = (\ell_j, i_j, t_j, x_j) \rightsquigarrow^\sharp_1 (\ell_{j+1}, i_{j+1}, t_{j+1}, x_{j+1})$ , for  $j \in \{1, 2\}$ . Then,  $\rightsquigarrow^\sharp_1 \triangleright \rightsquigarrow^\sharp_2$  where, for all  $pt_1, pt_2 \in Ports^\sharp(\Phi^\sharp)$ ,

$$pt_1 \rightsquigarrow^\sharp_2 pt_2 = \begin{cases} \overleftarrow{f_3}(\overrightarrow{f_1}(X_1) \sqcap_r^\sharp \overrightarrow{f_2}(X_2)) & \text{if } pt_1 = pt'_1 \text{ and } pt_2 = pt'_3 \\ \emptyset^\sharp & \text{otherwise} \end{cases}$$

The fiber  $\Delta\Phi^\sharp$  can be endowed with a lattice structure, the join  $\sqcup_{\Phi^\sharp}$  and the meet  $\sqcap_{\Phi^\sharp}$  being defined by componentwise application of  $\sqcup^\sharp$  and  $\sqcap^\sharp$ . We define similarly an operator  $\nabla_{\Phi^\sharp}$  on  $\Delta\Phi^\sharp$  by componentwise application of the widening operators provided by the abstract numerical domain.

**Proposition 7.**  $\nabla_{\Phi^\sharp}$  is a widening operator on  $\Delta\Phi^\sharp$ .

We define an endomorphism  $F_\triangleright$  on the fiber  $\Delta\Phi^\sharp$  as:

$$F_\triangleright \stackrel{\text{def}}{=} \lambda(\rightsquigarrow^\sharp) \cdot \bigsqcup_{\Phi^\sharp} \{ \rightsquigarrow^\sharp \bullet \mid \rightsquigarrow^\sharp \triangleright \rightsquigarrow^\sharp \bullet \}$$

**Definition 8 Abstract Closure.** Let  $\Phi^\sharp$  be an abstract distribution of processes and  $\rightsquigarrow^\sharp$  an element of  $\Delta\Phi^\sharp$ . The abstract closure  $[\rightsquigarrow^\sharp]_{\Phi^\sharp}$  of  $\rightsquigarrow^\sharp$  is given by the limit of the abstract iteration sequence of Theorem 4 applied to the function  $F_\triangleright$  on the domain  $\Delta\Phi^\sharp$ , with  $\rightsquigarrow^\sharp$  as abstract basis and  $\nabla_{\Phi^\sharp}$  as widening operator.  $\square$

Note that this closure operation enforces transitivity on an abstract channel relation  $\rightsquigarrow^\sharp$ . It may cause an important loss of accuracy, since the union of transitive relations (which is what is denoted by  $\rightsquigarrow^\sharp$ ) is in general not transitive. In fact, as we will see when we will define the abstract semantics, we only need to perform this closure operation on a relation  $\rightsquigarrow^\sharp = \rightsquigarrow^\sharp_1 \cup \rightsquigarrow^\sharp_2$  where  $\rightsquigarrow^\sharp_1$  denotes concrete channel relations and  $\rightsquigarrow^\sharp_2$  contains additional relations between ports in the domain of  $\rightsquigarrow^\sharp_1$  and some newly created ports. Therefore, it is possible to modify the rules defining  $\triangleright$  so that they can only be applied to pairs involving a new port, leaving the relation  $\rightsquigarrow^\sharp_1$  unchanged<sup>10</sup>. In order to keep the presentation simple, we do not detail this improvement of the analysis.

We define the abstract semantics of the  $\pi$ -calculus by a transition relation  $\Rightarrow^\sharp$  on  $\mathcal{D}^\sharp$  which mimics  $\Rightarrow$ . Let  $(\Phi^\sharp, \rightsquigarrow^\sharp)$  be an abstract configuration and  $\ell_o \in \mathcal{L}(S)$  such that:

$$S_{\ell_o} = \dots + \bar{z}[t_1, \dots, t_m] \cdot \nu v_1^o \dots \nu v_{i_o}^o \cdot (S_{\ell_1} \mid \dots \mid S_{\ell_{n_o}}) + \dots$$

We have a transition  $(\Phi^\sharp, \rightsquigarrow^\sharp) \Rightarrow^\sharp (\Phi^\sharp_\bullet, \rightsquigarrow^\sharp_\bullet)$  whenever there is  $\ell_i \in \mathcal{L}(S)$ , such that  $S_{\ell_i}$  is one of the following processes:

<sup>10</sup> This was pointed out to us by Alain Deutsch.

$$\begin{aligned}
& - \cdots + x(y_1, \dots, y_m) \cdot \nu v_1^i \dots \nu v_{t_i}^i \cdot (S_{\ell_1^i} \mid \cdots \mid S_{\ell_{t_i}^i}) + \cdots \\
& - !x(y_1, \dots, y_m) \cdot \nu v_1^i \dots \nu v_{t_i}^i \cdot (S_{\ell_1^i} \mid \cdots \mid S_{\ell_{t_i}^i})
\end{aligned}$$

and

$$(\ell_i, s_i, t_i, x) \rightsquigarrow^\# (\ell_o, s_o, t_o, z) \neq \emptyset^\#$$

for two abstract ports  $(\ell_i, s_i, t_i, x)$  and  $(\ell_o, s_o, t_o, z)$ . In other words, there is the possibility of a communication between two residuals of  $S_{\ell_i}$  and  $S_{\ell_o}$  in a concrete configuration abstracted by  $(\Phi^\#, \rightsquigarrow^\#)$ . We denote by  $X_c^\#$  the abstract value  $(\ell_i, s_i, t_i, x) \rightsquigarrow^\# (\ell_o, s_o, t_o, z)$ .

We define  $\Phi_\bullet^\#$  as follows:

$$\Phi_\bullet^\#(\ell) \stackrel{\text{def}}{=} \begin{cases} \Phi^\#(\ell) \oplus \mathcal{A}_\eta & \text{if } \ell = \ell_\eta^k, \text{ for } \eta \in \{i, o\}, k \in \{1, \dots, n_\eta\} \\ \Phi^\#(\ell) & \text{otherwise} \end{cases}$$

where  $\mathcal{A}_i$  and  $\mathcal{A}_o$  are automata representing the residuals of the newly spawned processes, and  $\oplus$  denotes the coproduct in  $\mathbb{A}$ . Whatever  $S_{\ell_i}$  may be, agent or server,  $\mathcal{A}_o$  is always defined as the automaton  $(Q_o, \{s_o\}, \{t_o\}, \tau_o)$ , where, for  $\eta \in \{i, o\}$ ,  $Q_\eta$  (resp.  $\tau_\eta$ ) is the set of states (resp. transitions) of  $\Phi^\#(\ell_\eta)$ . Indeed, in the nonstandard semantics the residuals of the processes spawned by the sender are always inherited from their parent agent. For  $\mathcal{A}_i$  there are two different cases depending on the type of  $S_{\ell_i}$ :

1.  $S_{\ell_i}$  is an agent. Then we put

$$\mathcal{A}_i \stackrel{\text{def}}{=} (Q_i, \{s_i\}, \{t_i\}, \tau_i)$$

2.  $S_{\ell_i}$  is a server. We suppose that we are provided with a distinguished element  $\Omega$  in the set  $\mathcal{Q}$  of all states of the automata in  $\mathbb{A}$ . Let  $Q_o \xrightarrow{\xi_1} Q_o \oplus \{\Omega\}$  and  $\{\Omega\} \xrightarrow{\xi_2} Q_o \oplus \{\Omega\}$  be the canonical inclusion maps in **FinSets**. We denote by  $\tau_o \xrightarrow{\Xi} \tau'_o$  the function that maps any transition  $(q, \ell, q')$  to  $(\xi_1(q), \ell, \xi_1(q'))$ . Then we put

$$\mathcal{A}_i \stackrel{\text{def}}{=} (Q_o \oplus \{\Omega\}, \{\xi_1(s_o)\}, \{\xi_2(\Omega)\}, \tau'_o)$$

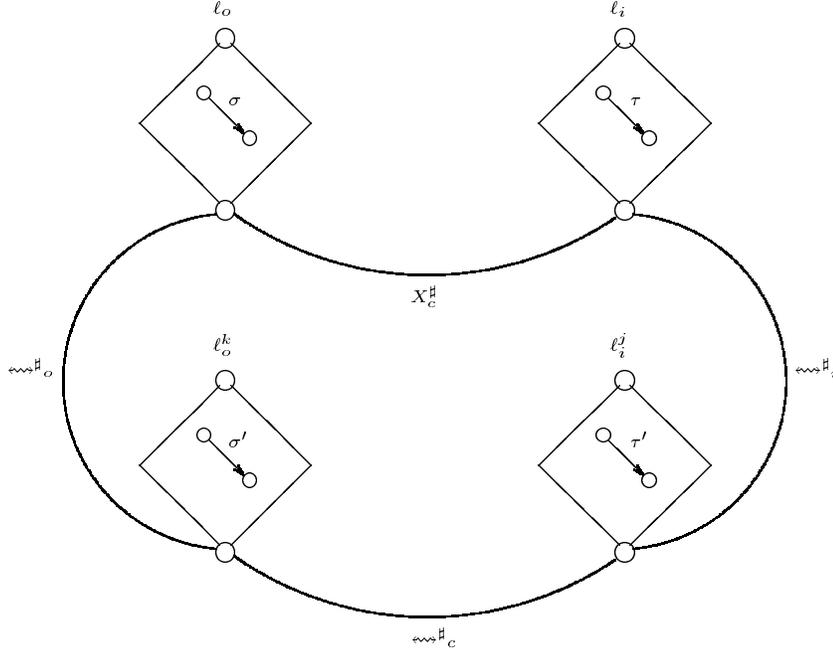
where  $\tau'_o \stackrel{\text{def}}{=} \{(\xi_1(q), \ell, \xi_1(q')) \mid (q, \ell, q') \in \tau_o\} \cup \{(\xi_1(t_o), \ell_o, \xi_2(\Omega))\}$ . This is the abstract counterpart of assigning the residual  $L_o \cdot \ell_o$  to the processes created by the server in the concrete semantics.

We denote by  $\mathcal{A}_\eta \xrightarrow{\iota_\eta} \Phi_\bullet^\#(\ell_\eta^l)$  the canonical inclusion morphism in  $\mathbb{A}$ , for  $\eta \in \{i, o\}$ ,  $l \in \{1, \dots, n_\eta\}$ . Note that the sets of residuals denoted by  $\Phi_\bullet^\#$  are larger than the ones denoted by  $\Phi^\#$ . Indeed, we are not allowed to remove anything since we have lost all information about the number of residuals of each  $S_{\ell_i}$ .

The abstract channel relation  $\rightsquigarrow_\bullet^\#$  mimics the concrete one:

$$\rightsquigarrow_\bullet^\# \stackrel{\text{def}}{=} [\Delta \mathcal{F}(\rightsquigarrow^\#) \sqcup_{\Phi_\bullet^\#} \rightsquigarrow_c^\# \sqcup_{\Phi_\bullet^\#} \rightsquigarrow_i^\# \sqcup_{\Phi_\bullet^\#} \rightsquigarrow_o^\# \sqcup_{\Phi_\bullet^\#} \rightsquigarrow_\nu^\#]_{\Phi_\bullet^\#}$$

where, for all  $\ell \in \mathcal{L}(S)$ ,  $\Phi^\#(\ell) \xrightarrow{\mathcal{F}(\ell)} \Phi_\bullet^\#(\ell)$  is the canonical inclusion morphism in  $\mathbb{A}$ . The definitions of  $\rightsquigarrow_c^\#$ ,  $\rightsquigarrow_i^\#$ ,  $\rightsquigarrow_o^\#$  and  $\rightsquigarrow_\nu^\#$  follow the lines of their concrete counterparts in Fig. 3 and Fig. 4. We distinguish two cases, depending on the type of  $S_{\ell_i}$ .



**Fig. 5.** Abstract interagent communication

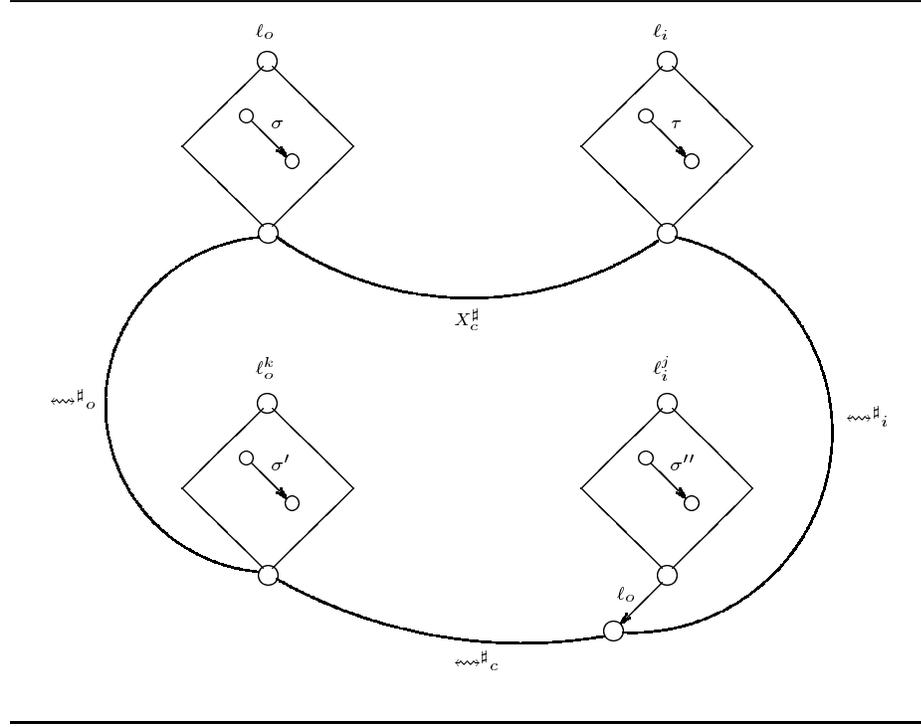
**Interagent Communication.** If  $S_{\ell_i}$  is an agent, the abstract semantics of communication is summarized in Fig. 5. We have represented the automata corresponding to the residuals of  $S_{\ell_i}$ ,  $S_{\ell_o}$ ,  $S_{\ell_i^j}$  and  $S_{\ell_o^k}$ , the latter being two processes spawned by the sender and the receiver after the communication occurred.

The abstract channel relation  $\rightsquigarrow_c^\sharp$  performs the binding between the names  $y_h$  and  $t_h$  involved in the communication. Since  $X_c^\sharp$  denotes residuals of  $S_{\ell_i}$  and  $S_{\ell_o}$  that may communicate, we use this same abstract numerical value to denote all possible channels between the ports associated to  $y_h$  and  $t_h$ . This can be stated formally as follows. For all  $j \in \{1, \dots, n_i\}$ ,  $k \in \{1, \dots, n_o\}$ ,  $h \in \{1, \dots, m\}$ ,  $y_h \in \text{fn}(S_{\ell_i^j})$  and  $t_h \in \text{fn}(S_{\ell_o^k})$ , we put

$$(\ell_i^j, (\ell_i^j)_1(s_i), (\ell_i^j)_1(t_i), y_h) \rightsquigarrow_c^\sharp (\ell_o^k, (\ell_o^k)_1(s_o), (\ell_o^k)_1(t_o), t_h) \stackrel{\text{def}}{=} \mathcal{V}^\sharp((\ell_i^j)_2 \oplus (\ell_o^k)_2)(X_c^\sharp)$$

For any other abstract ports  $pt_1, pt_2 \in \text{Ports}^\sharp(\Phi_\bullet^\sharp)$ , we put  $pt_1 \rightsquigarrow_c^\sharp pt_2 \stackrel{\text{def}}{=} \emptyset^\sharp$ .

The abstract channel relations  $\rightsquigarrow_i^\sharp$  and  $\rightsquigarrow_o^\sharp$  bind the free names in the instances of  $S_{\ell_i}$  and  $S_{\ell_o}$  to the corresponding ones in the processes released after the communication. Their rôle is thereby to propagate the scope of free names. The abstract numerical value associated to the relation  $\rightsquigarrow_i^\sharp$ , for example, is the



**Fig. 6.** Abstract agent-server communication

solution in the abstract numerical domain of the system of equations  $\tau = \tau'$ , where  $\tau$  is any transition in the automaton corresponding to  $S_{l_i}$  and  $\tau'$  is the copy of  $\tau$  in the automaton associated to the newly created process  $S_{l_i^j}$ , as shown in Fig. 5. The definition of  $\rightsquigarrow_o^\#$  is similar. The meaning of these operations at the concrete level is to bind pairs of ports with the same residuals. Since we abstract a residual by the number of times a path labelled by this residual runs through each transition of the automaton, our definition is quite natural. This can be stated more formally as follows. For all  $\eta \in \{i, o\}$ ,  $k \in \{1, \dots, n_\eta\}$  and  $u \in fn(S_{l_\eta}) \cap fn(S_{l_\eta^k})$ , we put

$$(\ell_\eta, \mathcal{F}(\ell_\eta)_1(s_\eta), \mathcal{F}(\ell_\eta)_1(t_\eta), u) \rightsquigarrow_\eta^\# (\ell_\eta^k, (\ell_\eta^k)_1(s_\eta), (\ell_\eta^k)_1(t_\eta), u) \stackrel{\text{def}}{=} \mathbf{Sol}^\#(\mathbf{S}_\eta^k)$$

where  $\mathbf{S}_\eta^k$  is the system of affine equations over  $\tau(\Phi_\bullet^\#(\ell_\eta)) \oplus \tau(\Phi_\bullet^\#(\ell_\eta^k))$  defined as follows:

$$\begin{cases} \zeta_1 \circ \mathcal{F}(\ell_\eta)_2(\sigma) = \zeta_2 \circ (\ell_\eta^k)_2(\sigma) \\ \sigma \in \tau_\eta \end{cases}$$

where  $\tau(\Phi_\bullet^\#(\ell_\eta)) \xrightarrow{\zeta_1} \tau(\Phi_\bullet^\#(\ell_\eta)) \oplus \tau(\Phi_\bullet^\#(\ell_\eta^k)) \xleftarrow{\zeta_2} \tau(\Phi_\bullet^\#(\ell_\eta^k))$  are the canonical inclusion maps in **FinSets**. For any other abstract ports  $pt_1, pt_2 \in Ports^\#(\Phi_\bullet^\#)$ , we put  $pt_1 \rightsquigarrow_\eta^\# pt_2 \stackrel{\text{def}}{=} \emptyset^\#$ .

The abstract channel relation  $\rightsquigarrow^\#_\nu$  binds the new ports created by the communication in the spawned processes. Since all processes released during a communication by either the sender or the receiver have the same residuals, we define  $\rightsquigarrow^\#_\nu$  similarly to  $\rightsquigarrow^\#_i$  and  $\rightsquigarrow^\#_o$  as the abstract solution of a system of linear equations. That is, for all  $\eta \in \{i, o\}$ ,  $j, k \in \{1, \dots, n_\eta\}$ , with  $j \neq k$ <sup>11</sup>, and  $j \in \{1, \dots, l_\eta\}$  such that  $v_\eta^j \in \text{fn}(S_{\ell_\eta^j}) \cap \text{fn}(S_{\ell_\eta^k})$ , we put

$$(\ell_\eta^j, (t_\eta^j)_1(s_\eta), (t_\eta^j)_1(t_\eta), u) \rightsquigarrow^\#_\nu (\ell_\eta^k, (t_\eta^k)_1(s_\eta), (t_\eta^k)_1(t_\eta), u) \stackrel{\text{def}}{=} \mathbf{Sol}^\#(\mathbf{S}_\eta^{j,k})$$

where  $\mathbf{S}_\eta^{j,k}$  is the system of affine equations over  $\tau(\Phi_\bullet^\#(\ell_\eta^j)) \oplus \tau(\Phi_\bullet^\#(\ell_\eta^k))$  defined as follows:

$$\begin{cases} \theta_1 \circ (t_\eta^j)_2(\sigma) = \theta_2 \circ (t_\eta^k)_2(\sigma) \\ \sigma \in \tau_\eta \end{cases}$$

where  $\tau(\Phi_\bullet^\#(\ell_\eta^j)) \xrightarrow{\theta_1} \tau(\Phi_\bullet^\#(\ell_\eta^j)) \oplus \tau(\Phi_\bullet^\#(\ell_\eta^k)) \xleftarrow{\theta_2} \tau(\Phi_\bullet^\#(\ell_\eta^k))$  are the canonical inclusion maps in **FinSets**. For any other abstract ports  $pt_1, pt_2 \in \text{Ports}^\#(\Phi_\bullet^\#)$ , we put  $pt_1 \rightsquigarrow^\#_\nu pt_2 \stackrel{\text{def}}{=} \emptyset^\#$ .

**Agent-Server Communication.** If  $S_{\ell_i}$  is a server, the abstract semantics of communication is summarized in Fig. 6. The abstract channel relations  $\rightsquigarrow^\#_o$  and  $\rightsquigarrow^\#_\nu$  are constructed exactly as above. The situation is quite different for  $\rightsquigarrow^\#_i$  and  $\rightsquigarrow^\#_c$ , because the automata corresponding to the processes spawned by  $S_{\ell_i}$  are copies of the automaton associated to  $S_{\ell_o}$  suffixed with a transition labelled by  $\ell_o$ .

The definition of  $\rightsquigarrow^\#_c$  is very similar to the definition of  $\rightsquigarrow^\#_i$  in the previous case. The binding between the names involved in the communication is performed by taking the solution of the system of equations  $\sigma' = \sigma''$  in the abstract numerical domain, as illustrated in Fig. 6. Since residuals of the processes released by the receiver are suffixed with  $\ell_o$ , we furthermore require that the corresponding transition be crossed exactly once. This operation can be formalized as follows. For all  $j \in \{1, \dots, n_i\}$ ,  $k \in \{1, \dots, n_o\}$ ,  $h \in \{1, \dots, m\}$ ,  $y_h \in \text{fn}(S_{\ell_i^j})$  and  $t_h \in \text{fn}(S_{\ell_o^k})$ , we put

$$(\ell_i^j, (t_i^j)_1 \circ \xi_1(s_o), (t_i^j)_1 \circ \xi_2(\Omega), y_h) \rightsquigarrow^\#_c (\ell_o^k, (t_o^k)_1(s_o), (t_o^k)_1(t_o), t_h) \stackrel{\text{def}}{=} \mathbf{Sol}^\#(\mathbf{S}_c^{j,k})$$

where  $\mathbf{S}_c^{j,k}$  is the system of affine equations over  $\tau(\Phi_\bullet^\#(\ell_i^j)) \oplus \tau(\Phi_\bullet^\#(\ell_o^k))$  defined as follows:

$$\begin{cases} \zeta_1 \circ (t_i^j)_2(\xi_1(t_o), \ell_o, \xi_2(\Omega)) = 1 \\ \zeta_1 \circ (t_i^j)_2 \circ \Xi(\sigma) = \zeta_2 \circ (t_o^k)_2(\sigma) \\ \sigma \in \tau_o \end{cases}$$

where  $\tau(\Phi_\bullet^\#(\ell_i^j)) \xrightarrow{\zeta_1} \tau(\Phi_\bullet^\#(\ell_i^j)) \oplus \tau(\Phi_\bullet^\#(\ell_o^k)) \xleftarrow{\zeta_2} \tau(\Phi_\bullet^\#(\ell_o^k))$  are the canonical inclusion maps in **FinSets**. For any other abstract ports  $pt_1, pt_2 \in \text{Ports}^\#(\Phi_\bullet^\#)$ , we put  $pt_1 \rightsquigarrow^\#_c pt_2 \stackrel{\text{def}}{=} \emptyset^\#$ .

<sup>11</sup> Reflexivity is implicit by definition of  $\gamma$ .

The definition of  $\rightsquigarrow^\#_i$  is slightly more delicate. The residuals of the processes spawned by the server are represented by a copy of the automaton associated to the sender and  $\rightsquigarrow^\#_i$  relates the instances of  $S_{\ell_i}$  which may communicate with  $S_{\ell_o}$  to the instances of  $S_{\ell_i^j}$  which have been effectively spawned by the communication. Thereby, the abstract numerical value associated to  $\rightsquigarrow^\#_i$  consists of  $X_c^\#$  for which each  $\sigma$  has been mapped to its copy  $\sigma''$  in the automaton corresponding to  $S_{\ell_i^j}$  (see Fig. 6), together with the constraint that the additional transition labelled by  $\ell_o$  must be crossed once. This can be stated formally as follows. Let  $\tau'_i \stackrel{\text{def}}{=} \tau_i \oplus \tau'_o$  and  $\tau_i \xrightarrow{\chi_1} \tau'_i, \tau'_o \xrightarrow{\chi_2} \tau'_i$  be the canonical inclusion maps in **FinSets**. For all  $k \in \{1, \dots, n_i\}$  and  $u \in \text{fn}(S_{\ell_i}) \cap \text{fn}(S_{\ell_i^k})$ , we put

$$(l_i, \mathcal{F}(l_i)_1(s_i), \mathcal{F}(l_i)_1(t_i), u) \rightsquigarrow^\#_i (\ell_i^k, (t_i^k)_1 \circ \xi_1(s_o), (t_i^k)_1 \circ \xi_2(\Omega), u) \stackrel{\text{def}}{=} \mathcal{V}^\#(\mathcal{F}(l_i)_2 \oplus (t_i^k)_2) \left( \overrightarrow{[\chi_1 \oplus (\chi_2 \circ \Xi)]} (X_c^\#) \cap \tau_i^\# \mathbf{Sol}_{\tau_i^\#}^\#(\{\chi_2(\xi_1(t_o), \ell_o, \xi_2(\Omega)) = 1\}) \right)$$

For any other abstract ports  $pt_1, pt_2 \in \text{Ports}^\#(\Phi_\bullet^\#)$ , we put  $pt_1 \rightsquigarrow^\#_i pt_2 \stackrel{\text{def}}{=} \emptyset^\#$ .

**Definition 9 Abstract Semantic Function.** Let  $c^\# = (\Phi^\#, \rightsquigarrow^\#)$  be an abstract configuration. For any configuration  $(\Phi_\bullet^\#, \rightsquigarrow^\#_\bullet)$  such that  $c^\# \Rightarrow^\# (\Phi_\bullet^\#, \rightsquigarrow^\#_\bullet)$ , we denote by  $\mathcal{F}[\Phi_\bullet^\#, \rightsquigarrow^\#_\bullet] : \Phi^\# \longrightarrow \Phi_\bullet^\#$  the canonical inclusion morphism. Let  $\Phi_\dagger^\#$  be the colimit in  $\Phi^\#(S)$  of the diagram  $\{\mathcal{F}[\Phi_\bullet^\#, \rightsquigarrow^\#_\bullet] : \Phi^\# \longrightarrow \Phi_\bullet^\# \mid c^\# \Rightarrow^\# (\Phi_\bullet^\#, \rightsquigarrow^\#_\bullet)\}$  and  $\{\mathcal{F}^\dagger[\Phi_\bullet^\#, \rightsquigarrow^\#_\bullet] : \Phi^\# \longrightarrow \Phi_\dagger^\# \mid c^\# \Rightarrow^\# (\Phi_\bullet^\#, \rightsquigarrow^\#_\bullet)\}$  be the colimiting cocone. We define the abstract semantic function  $F^\# : \mathcal{D}^\# \longrightarrow \mathcal{D}^\#$  as follows:

$$F^\#(c^\#) \stackrel{\text{def}}{=} \bigsqcup_{\Phi_\dagger^\#} \{\Delta \mathcal{F}^\dagger[\Phi_\bullet^\#, \rightsquigarrow^\#_\bullet](\rightsquigarrow^\#_\bullet) \mid c^\# \Rightarrow^\# (\Phi_\bullet^\#, \rightsquigarrow^\#_\bullet)\}$$

□

**Theorem 10 Soundness of the Abstract Semantics.**

$$F_S \circ \gamma \subseteq \gamma \circ F^\#$$

It now remains to define a widening operator on  $\mathcal{D}^\#$  in order to make the abstract semantics of  $S$  computable.

## 7 Widening Operators

There is a generic method to construct widening operators on cofibered domains by combining the widenings defined locally on each fiber [Ven96]. This requires to extend the notion of widening to categories.

**Definition 11 Widening on a Category.** Let  $\mathbb{C}$  be a category. A *widening operator*  $\nabla$  on  $\mathbb{C}$  associates to any two objects  $X, Y$  of  $\mathbb{C}$  two arrows:

$$X \xrightarrow{X \vec{\nabla}_1 Y} X \nabla Y \xleftarrow{X \vec{\nabla}_2 Y} Y$$

such that for any sequence of objects  $(X_n)_{n \geq 0}$ , the sequence of arrows  $(X_n \nabla \xrightarrow{f_n^\nabla} X_{n+1}^\nabla)_{n \geq 0}$  in  $\mathbb{C}$  defined by:

$$\begin{cases} X_0^\nabla = X_0 \\ X_{n+1}^\nabla = X_n^\nabla \nabla X_{n+1} \\ f_n^\nabla = X_n^\nabla \vec{\nabla}_1 X_{n+1} \end{cases}$$

is ultimately pseudo-stationary, i.e. there exists  $N$  such that, for all  $n \geq N$ ,  $f_n^\nabla$  is an isomorphism. Moreover we require  $\nabla$  to be stable under isomorphism, that is, whenever  $X \cong X'$  and  $Y \cong Y'$ , then  $X \nabla Y \cong X' \nabla Y'$ .  $\square$

Assuming that  $\Phi^\sharp(S)$  is provided with a widening  $\nabla$ , we construct a widening operator  $\nabla^\sharp$  on  $\mathcal{D}^\sharp$  as follows. Let  $(\Phi_1^\sharp, \rightsquigarrow^\sharp_1)$  and  $(\Phi_2^\sharp, \rightsquigarrow^\sharp_2)$  be abstract configurations,  $(\Phi_1^\sharp, \rightsquigarrow^\sharp_1) \nabla^\sharp (\Phi_2^\sharp, \rightsquigarrow^\sharp_2)$  is given by:

- $(\Phi_1^\sharp, \rightsquigarrow^\sharp_1 \nabla_{\Phi_1^\sharp} \Delta((\Phi_1^\sharp \vec{\nabla}_1 \Phi_2^\sharp)^{-1} \circ (\Phi_1^\sharp \vec{\nabla}_2 \Phi_2^\sharp))(\rightsquigarrow^\sharp_2))$  when  $\Phi_1^\sharp \vec{\nabla}_1 \Phi_2^\sharp$  is an isomorphism,
- $(\Phi_1^\sharp \nabla \Phi_2^\sharp, \Delta(\Phi_1^\sharp \vec{\nabla}_1 \Phi_2^\sharp)(\rightsquigarrow^\sharp_1) \nabla_{\Phi_1^\sharp \nabla \Phi_2^\sharp} \Delta(\Phi_1^\sharp \vec{\nabla}_2 \Phi_2^\sharp)(\rightsquigarrow^\sharp_2))$  otherwise.

Intuitively the first case means that when the fiber is “stable”, i.e.  $\Phi_1^\sharp \vec{\nabla}_1 \Phi_2^\sharp$  is an isomorphism, we “transfer” the abstract channel relation  $\rightsquigarrow^\sharp_2$  into the fiber and we make the widening with  $\rightsquigarrow^\sharp_1$ :

$$\begin{array}{ccc} \Phi_1^\sharp & \xleftarrow{(\Phi_1^\sharp \vec{\nabla}_1 \Phi_2^\sharp)^{-1}} & \Phi_1^\sharp \nabla \Phi_2^\sharp \xleftarrow{\Phi_1^\sharp \vec{\nabla}_2 \Phi_2^\sharp} \Phi_2^\sharp \\ | & & | \\ \nabla_{\Phi_1^\sharp} \left\{ \begin{array}{l} \rightsquigarrow^\sharp_1 \\ \Delta((\Phi_1^\sharp \vec{\nabla}_1 \Phi_2^\sharp)^{-1} \circ (\Phi_1^\sharp \vec{\nabla}_2 \Phi_2^\sharp))(\rightsquigarrow^\sharp_2) \end{array} \right. & \xleftarrow{\dots\dots\dots} & \rightsquigarrow^\sharp_2 \end{array}$$

Otherwise we transfer  $\rightsquigarrow^\sharp_1$  and  $\rightsquigarrow^\sharp_2$  into the fiber over  $\Phi_1^\sharp \nabla \Phi_2^\sharp$  and we make the widening in this fiber:

$$\begin{array}{ccc} \Phi_1^\sharp & \xrightarrow{\Phi_1^\sharp \vec{\nabla}_1 \Phi_2^\sharp} & \Phi_1^\sharp \nabla \Phi_2^\sharp \xleftarrow{\Phi_1^\sharp \vec{\nabla}_2 \Phi_2^\sharp} \Phi_2^\sharp \\ | & & | \\ \rightsquigarrow^\sharp_1 \dashrightarrow \Delta(\Phi_1^\sharp \vec{\nabla}_1 \Phi_2^\sharp)(\rightsquigarrow^\sharp_1) \nabla_{\Phi_1^\sharp \nabla \Phi_2^\sharp} \Delta(\Phi_1^\sharp \vec{\nabla}_2 \Phi_2^\sharp)(\rightsquigarrow^\sharp_2) \dashleftarrow & & \rightsquigarrow^\sharp_2 \end{array}$$

**Theorem 12.**  $\nabla^\sharp$  is a widening operator on  $(\mathcal{D}^\sharp, \preceq)$ .

We now define a widening  $\nabla$  on  $\Phi^\sharp(S)$  by componentwise application of a widening  $\nabla_{\mathbb{A}}$  on  $\mathbb{A}$ . The idea is to fold states in an automaton  $\mathcal{A} = (Q, I, T, \tau)$  with respect to some “similarity” criterion which is represented by an equivalence relation  $\equiv_{\mathcal{A}}^{\nabla}$  on  $Q$ . The folding operation is achieved by quotienting the automaton with respect to this relation.

*Example 7.* Let  $k$  be a nonnegative integer. For any  $q \in Q$ , we denote by  $L_{\leq k}(q)$  the set of words labelling a path in  $\mathcal{A}$  originating from  $q$  and of length bounded by  $k$ . We define  $\equiv_{\mathcal{A}}^{\nabla}$  as follows:

$$q \equiv_{\mathcal{A}}^{\nabla} q' \iff L_{\leq k}(q) = L_{\leq k}(q')$$

This folding criterion is inspired by the *k-limiting* approximation of [JM81].  $\square$

**Definition 13 Quotient of an Automaton.** Let  $\mathcal{A} = (Q, I, T, \tau)$  be an automaton of  $\mathbb{A}$  and  $\sim$  be an equivalence relation on  $Q$ . We denote by  $\pi_{\sim} : Q \longrightarrow Q/\sim$  the canonical projection onto the quotient set<sup>12</sup>. The quotient  $\mathcal{A}/\sim$  of  $\mathcal{A}$  by  $\sim$  is then defined as

$$\mathcal{A}/\sim \stackrel{\text{def}}{=} (Q/\sim, \pi_{\sim}(I), \pi_{\sim}(T), \{(\pi_{\sim}(q), \ell, \pi_{\sim}(q')) \mid (q, \ell, q') \in \tau\})$$

$\square$

*Example 8.* Let  $\equiv_{\delta}$  be the smallest equivalence relation on  $Q$  such that:

1.  $\forall i, i' \in I : i \equiv_{\delta} i'$ .
2.  $\forall (q, \ell, r), (q', \ell', r') \in \tau : q \equiv_{\delta} q' \wedge \ell = \ell' \implies r \equiv_{\delta} r'$ .

Then it is obvious that the quotient automaton  $\mathcal{A}/\equiv_{\delta}$  is deterministic. This is an approximate<sup>13</sup> procedure to make an automaton deterministic.  $\square$

Let  $\mathcal{A}_1, \mathcal{A}_2$  be two automata. We denote by  $\mathcal{A}_i \xrightarrow{e_i} \mathcal{A}_1 \oplus \mathcal{A}_2$ ,  $i \in \{1, 2\}$ , the canonical inclusion morphisms in  $\mathbb{A}$ . If for each automaton  $\mathcal{A}$  in  $\mathbb{A}$  we have a computable equivalence relation  $\equiv_{\mathcal{A}}^{\nabla}$  on the states of  $\mathcal{A}$ , we put

$$\mathcal{A}_1 \nabla_{\mathbb{A}} \mathcal{A}_2 \stackrel{\text{def}}{=} (\mathcal{A}_1 \oplus \mathcal{A}_2) / \equiv_{\mathcal{A}_1 \oplus \mathcal{A}_2}^{\nabla}$$

If  $\mathcal{A}_1 \oplus \mathcal{A}_2 \xrightarrow{\pi} (\mathcal{A}_1 \oplus \mathcal{A}_2) / \equiv_{\mathcal{A}_1 \oplus \mathcal{A}_2}^{\nabla}$  is the morphism of automata induced by the canonical projection  $\pi_{\equiv_{\mathcal{A}_1 \oplus \mathcal{A}_2}^{\nabla}}$ , we put  $\mathcal{A}_i \xrightarrow{\nabla_i} \mathcal{A}_2 \stackrel{\text{def}}{=} \pi \circ e_i$ , for  $i \in \{1, 2\}$ .

We cannot say much about  $\nabla_{\mathbb{A}}$  in whole generality. We have to check by hand that each operator defined in this way satisfies the properties of Definition 11.

**Proposition 14.** *The operator  $\nabla_{\mathbb{A}}$  defined from the equivalence relations  $\equiv_{\mathcal{A}}^{\nabla}$  of Example 7 is a widening on  $\mathbb{A}$ .*

<sup>12</sup> Strictly speaking  $Q/\sim$  is a representative of the quotient set in  $\mathcal{Q}$ .

<sup>13</sup> The language recognized by the quotient automaton can be larger than the original one.

It only remains to define the abstract basis  $\perp^\# = (\Phi_\perp^\#, \rightsquigarrow_\perp^\#)$  in order to compute the iteration sequence. If  $S$  is the labelled  $\pi$ -term  $\nu x_1 \dots \nu x_m. (\ell_1 : AS_1 \mid \dots \mid \ell_n : AS_n)$ , where the  $AS_i$  are either agents or servers,  $\Phi_\perp^\#$  is defined as follows:

$$\Phi_\perp^\#(\ell) \stackrel{\text{def}}{=} \begin{cases} (\{\Omega\}, \{\Omega\}, \{\Omega\}, \emptyset) & \text{if } \ell \in \{\ell_1, \dots, \ell_n\} \\ (\emptyset, \emptyset, \emptyset, \emptyset) & \text{otherwise} \end{cases}$$

For all distinct  $\ell, \ell' \in \{\ell_1, \dots, \ell_n\}$  and all  $x \in \text{fn}(S_\ell) \cap \text{fn}(S_{\ell'})$ , we put

$$(\ell, \Omega, \Omega, x) \rightsquigarrow_\perp^\# (\ell', \Omega, \Omega, x) \stackrel{\text{def}}{=} \top^\#$$

*Example 9.* If we use the widening of Example 7 and enforce the automata to be deterministic with the procedure of Example 8, the limit of the abstract iteration sequence corresponding to the  $\pi$ -term  $S$  of Example 2 is given by the abstract configuration defined in Example 5. Our analysis is thus able to infer non-uniform communication topologies.  $\square$

## 8 Conclusion

We have presented an analysis of communications in the  $\pi$ -calculus based on Abstract Interpretation. This analysis is able to infer accurate descriptions of the communication topology of a system  $S$  of mobile processes, since it can distinguish instances of recursively spawned processes, as illustrated by Examples 5 and 9. This can be applied in particular to problems of security in distributed systems, where this kind of information is required to ensure that confidential data cannot be accessed by unauthorized elements of the system. Another possible application would be to use the information about the communication topology to statically derive an allocation strategy of the processes on a multiprocessor architecture which optimizes the communication cost. Future work will focus on the implementation of the analysis and the experimental study of its usefulness for these applications.

Further extensions to the abstract interpretation of the  $\pi$ -calculus are to be investigated. One can enrich the abstract domain to take into account the number of instances of a process. This may improve the analysis and give interesting information for compilation (channels used only once, finite communication topologies, etc.). However the major extension is to consider  $\pi$ -terms with free variables in order to achieve modular analysis. This requires to modify the non-standard semantics in order to model interaction with the environment. Finally, other directions for the approximation of the communication topology should be studied, like hypergraph grammars in the style of [Hab92].

**Acknowledgements:** I am grateful to Torben Amtoft, Radhia Cousot, Patrick Cousot and Ian Mackie for helpful comments on first versions of this paper. All diagrams have been designed using Paul Taylor's and Paul Gastin's  $\text{\LaTeX}$  packages.

## References

- [BB92] G. Berry and G. Boudol. The chemical abstract machine. *Theoretical Computer Science*, 96:217–248, 1992.
- [BCHK94] G. Boudol, I. Castellani, M. Hennessy, and A. Kiehn. A theory of processes with localities. *Formal Aspects of Computing*, 6(2):165–200, 1994.
- [BW90] M. Barr and C. Wells. *Category Theory for Computing Science*. Prentice Hall, 1990.
- [CC76] P. Cousot and R. Cousot. Static determination of dynamic properties of programs. In *Proceedings of the 2<sup>nd</sup> International Symposium on Programming*, pages 106–130, Paris, 1976. Dunod.
- [CC77] P. Cousot and R. Cousot. Abstract interpretation : a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Conference Record of the 4<sup>th</sup> ACM Symposium on Principles of Programming Languages*, pages 238–252, Los Angeles, California, U.S.A., 1977.
- [CC92] P. Cousot and R. Cousot. Abstract interpretation frameworks. *Journal of logic and computation*, 2(4):511–547, August 1992.
- [CC95] P. Cousot and R. Cousot. Compositional and inductive semantic definitions in fixpoint, equational, constraint, closure-condition, rule-based and game theoretic form. In *Conference on Computer-Aided Verification, 7th International Conference, CAV'95*, volume 939 of *Lecture Notes in Computer Science*, pages 293–308. Springer-Verlag, 1995. Invited paper.
- [CH78] P. Cousot and N. Halbwachs. Automatic discovery of linear restraints among variables of a program. In *5<sup>th</sup> POPL*. ACM Press, 1978.
- [Col95a] C. Colby. Analyzing the communication topology of concurrent programs. In *Symposium on Partial Evaluation and Program Manipulation*, 1995.
- [Col95b] C. Colby. Determining storage properties of sequential and concurrent programs with assignment and structured data. In *Proceedings of the Second International Static Analysis Symposium*, volume 983 of *Lecture Notes in Computer Science*, pages 64–81. Springer-Verlag, 1995.
- [Deu92a] A. Deutsch. *Operational models of programming languages and representations of relations on regular languages with application to the static determination of dynamic aliasing properties of data*. PhD thesis, University Paris VI (France), 1992.
- [Deu92b] A. Deutsch. A storeless model of aliasing and its abstraction using finite representations of right-regular equivalence relations. In *Proceedings of the 1992 International Conference on Computer Languages*, pages 2–13. IEEE Computer Society Press, Los Alamitos, California, U.S.A., 1992.
- [Deu94] A. Deutsch. Interprocedural may-alias analysis for pointers : beyond k-limiting. In *ACM SIGPLAN'94 Conference on Programming Language Design and Implementation*. ACM Press, 1994.
- [Gra89] P. Granger. Static analysis of arithmetical congruences. *International Journal of Computer Mathematics*, 30:165–190, 1989.
- [Gra91] P. Granger. Static analysis of linear congruence equalities among variables of a program. In *TAPSOFT'91*, volume 493. Lecture Notes in Computer Science, 1991.
- [Hab92] A. Habel. *Hyperedge replacement: grammars and languages*, volume 643 of *Lecture Notes in Computer Science*. Springer-Verlag, 1992.

- [JM81] N. Jones and S. Muchnick. Flow analysis and optimization of lisp-like structures. In *Program Flow Analysis: Theory and Applications*, pages 102–131. Prentice Hall, 1981.
- [Jon81] H.B.M Jonkers. Abstract storage structures. In De Bakker and Van Vliet, editors, *Algorithmic languages*, pages 321–343. IFIP, 1981.
- [Kar76] M. Karr. Affine relationships among variables of a program. *Acta Informatica*, pages 133–151, 1976.
- [Mil91] R. Milner. The polyadic  $\pi$ -calculus: a tutorial. In *Proceedings of the International Summer School on Logic and Algebra of Specification*. Springer Verlag, 1991.
- [Mil92] R. Milner. Functions as processes. *Journal of Mathematical Structures in Computer Science*, 2, 1992.
- [MPW92] R. Milner, J. Parrow, and D. Walker. A calculus of mobile processes. *Information and Computation*, 100:1 – 77, 1992.
- [NN94] H. R. Nielson and F. Nielson. Higher-order concurrent programs with finite communication topology. In 21<sup>st</sup> *ACM Symposium on Principles of Programming Languages*, 1994.
- [San94a] D. Sangiorgi. *Expressing mobility in process algebras: first-order and higher-order paradigms*. PhD thesis, University of Edinburgh, 1994.
- [San94b] D. Sangiorgi. Locality and true-concurrency in calculi for mobile processes. In *International Symposium on Theoretical Aspects of Computer Software*, Lecture Notes in Computer Science, 1994.
- [Tur95] D. N. Turner. *The Polymorphic Pi-Calculus: Theory and Implementation*. PhD thesis, Edinburgh University, 1995.
- [Ven96] A. Venet. Abstract cofibered domains: Application to the alias analysis of untyped programs. In *Proc. of the Third International Static Analysis Symposium SAS'96*, volume 1145 of *Lecture Notes in Computer Science*, pages 366–382. Springer-Verlag, 1996.