On Abstract Interpretation of Mobile Ambients [*]

Francesca Levi
DISI, University of Genova, Italy

Sergio Maffeis
Department of Computing, Imperial College, UK

leviﬁran@disi.unige.it
maffeis@doc.ic.ac.uk

April 14, 2003

Abstract

We introduce an abstract interpretation framework for Mobile Ambients, based on a new semantics called *normal semantics*. Then, we derive within this setting two analyses computing a safe approximation of the run-time topological structure of processes. Such a static information can be successfully used to establish interesting security properties.

1 Introduction

Mobile Ambients (MA) [10] has recently emerged as a core programming language for the Web, and at the same time as a model for reasoning about properties of mobile processes. MA is based on the notion of *ambient*. An ambient is a bounded place, where multi-threaded computation takes place; roughly speaking, it generalises both the idea of agent and the idea of location. Each ambient has a *name*, a collection of *local processes* and a collection of *subambients*. Ambients are organised in a tree, which can be dynamically modified, according to three basic capabilities: $\mathtt{in}\, n$ allows an ambient to enter into an ambient n ($m[\mathtt{in}\, n. P_1 \mid P_2] \mid n[Q] \to n[m[P_1 \mid P_2] \mid Q]$); $\mathtt{out}\, n$ allows an ambient to exit from an ambient n ($n[m[\mathtt{out}\, n. P_1 \mid P_2] \mid Q] \to m[P_1 \mid P_2] \mid n[Q]$); $\mathtt{open}\, n$ allows to destroy the boundary of an ambient n ($\mathtt{open}\, n. P \mid n[Q] \to P \mid Q$).

Several static techniques, formalised as Type Systems [20, 8, 7, 9, 17, 6, 2, 3, 11, 19] or Control Flow Analyses (CFA) in Flow Logic style [24, 25, 26, 16, 5], have been devised

[*]The work has been partially done when the authors were at the *Department of Computer Science* of the University of Pisa. They have been partially supported by the MURST project *Abstract Interpretation, Type Systems and Control Flow Analysis.*

to study and establish various security properties of MA, such as secrecy and information flow. These approaches are strictly related and compute safe approximations of similar information on the run-time topological structure of processes. Although these methods are proved sound with respect to a formal semantics, they are typically formulated in different styles. As a consequence, it is rather difficult to formally compare them, and the corresponding algorithms for constructing the least analysis or for type-inference.

In this paper we apply to MA the semantic-based approach to program analysis of *abstract interpretation* [14, 13]. Abstract interpretation provides a rigorous theory to derive program analyses from the specification of the semantics. The typical abstract interpretation approach consists of: replacing the concrete domain of computation with an abstract domain modeling the property we are interested in; establishing a relation between the concrete and the abstract domain which formalises (through Galois connections) safeness and precision of approximations; deriving an approximate semantics over the abstract domain. The approximate semantics can be obtained in a systematic way which guarantees its safeness by construction. We refer the reader to Section 2 for more details on the basic concepts of the abstract interpretation theory.

One of the most important and critical steps for applying abstract interpretation consists of the choice of the concrete semantics one should start from. The standard reduction semantics of MA [10] is not adequate to abstraction, because it heavily relies on the syntax by using structural rules and structural congruence to bring the participants of a potential reaction into contiguous positions. We therefore introduce a new semantics for MA, called *normal semantics*, which is indeed equivalent to the standard reduction semantics. The normal semantics is based on the simple observation that an MA process is essentially a tree, where each node is an ambient containing a set of local processes controlling its movements. Then we derive, by step-wise abstraction of the normal semantics, two analyses which are proved to be safe.

The first analysis is designed to compute an approximation of the following property of all the computations of a process P: for any ambient n, which ambients and capabilities may be contained (at top level) inside n, when n is within an ambient h. This is obtained by an abstraction which combines information about the number of occurrences of objects and about the context. The integration of these two aspects permits to achieve very accurate results. To substantiate this claim, we consider a typical example: an ambient n which moves inside an immobile ambient k, and then is opened unleashing an immobile process inside k. This kind of situation is critical in MA, if we want to prove statically the immobility of k, as it is necessary to detect that any capability of movement inside n has been consumed *before* opening. Example 5.11 shows that our analysis achieves this result, in particular because it is able to argue on the temporal ordering of execution of capabilities. We are not aware of similar results in the setting of MA without adopting more complex techniques [26, 1]. It is well-know instead that this problem can be solved with simpler techniques for variants of MA, such as Safe Ambients (SA) [20, 21]. The static techniques for SA [21, 20, 16, 2, 17, 19] are typically more precise due to the presence of coactions, which control when an interaction may happen. For instance, the coaction $\overline{\texttt{open}}$

simplifies the task of distinguishing what happens inside an ambient *before* and *after* it is opened. Similar results has been obtained also for MA extended with primitives for objective mobility [7].

The second analysis is designed to compute an approximation of the following weaker property of all the computations of a process P: for any ambient n, which ambients and capabilities may be contained (at top level) inside n. This is obtained from the first analysis by dropping off both the contextual information and the information about the number of occurrences of objects. The analysis we obtain is a refined version of the CFA of [24]. The main difference with respect to [24] is that our analysis considers the effect of the continuation of a capability only if the capability may be exercised. Example 6.11 shows in details the difference with the CFA of [24].

The properties computed by both the analyses permit to control where an ambient may move and also where it may be opened. This is the basic information which is needed to statically establish most of the security properties studied in the literature for MA [5, 6, 9, 16, 17, 24]. To illustrate the relevance of the analysis for security we show the application to some well-known examples taken from [16, 5]. We focus on the first analysis which is more precise and interesting; the second analysis can be used, as the CFA of [24], to solve simpler problems, such as the firewall protocol of [10] and the Trojan Horse of [6].

The normal semantics is presented in Section 4, and the two derived abstractions in Sections 5 and 6, respectively. Section 7 shows some examples of security properties. The proof of the main theorems are shown in the Appendixes A and B.
Remark This paper is an extended and revised version of [22].

2 Some background on abstract interpretation

We briefly recall the basic concepts of the Galois connection based approach of abstract interpretation [14, 13]. Suppose we want to approximate a semantics $\mathcal{S}$, which is computed as the least fixed-point of a monotonic function F over some *concrete domain* $\langle C, \leq \rangle$. The key step consists of the choice of an *abstract domain* $\langle A, \leq^\alpha \rangle$ modeling the property we want to statically establish. The notion of Galois connection formalises the relation of abstraction between the concrete and the abstract domain which is the basis to define safeness and precision of approximations.

Definition 2.1 (Galois connection) *Let $\langle C, \leq \rangle$ and $\langle A, \leq^\alpha \rangle$ be complete lattices. A pair of monotonic functions (α, γ), such that $\alpha : C \to A$ is the* abstraction *function and $\gamma : A \to C$ is the* concretization *function, is a* Galois connection *between $\langle C, \leq \rangle$ and $\langle A, \leq^\alpha \rangle$ iff, for each $c \in C$ and $a \in A$*

1. $c \leq \gamma(\alpha(c))$;
2. $\alpha(\gamma(a)) \leq^\alpha a$.

When $\alpha(\gamma(a)) = a$, then (α, γ) is called a Galois insertion.

The ordering $\leq^\alpha$ is intended to model precision so that $a \leq^\alpha a'$ means that a' is a safe approximation of a. Therefore, the abstraction of the least fixed-point $\alpha(\mathcal{S})$ gives the exact abstract property corresponding to $\mathcal{S}$, and an approximate semantics $\mathcal{S}^\alpha$ over the abstract domain is a *safe approximation* of $\mathcal{S}$ whenever $\alpha(\mathcal{S}) \leq^\alpha \mathcal{S}^\alpha$. One of the main results of abstract interpretation is that a safe approximate semantics $\mathcal{S}^\alpha$ can be computed as the least fixed-point of an abstract function F^α satisfying a condition of local safeness, namely that $\alpha(F(c)) \leq^\alpha F^\alpha(\alpha(c))$.

Theorem 2.2 (Safeness) *Let (α, γ) be a Galois connection between $\langle C, \leq\rangle$ and $\langle A, \leq^\alpha\rangle$. Moreover, let $F : C \to C$ and $F^\alpha : A \to A$ be monotonic functions. If $\alpha(F(c)) \leq^\alpha F^\alpha(\alpha(c))$, for each $c \in C$, then $\alpha(lfp\ F) \leq^\alpha lfp\ F^\alpha$.*

3 Mobile Ambients

We introduce the pure *Mobile Ambients* calculus ([10]) without communication primitives. Let $\mathcal{N}$ be a set of *names* (ranged over by $n, m, h, k, \ldots$).

Definition 3.1 (Processes) *The* processes *are defined over names $\mathcal{N}$ according to the following syntax:*

M,N::=		*(capabilities)*	*P,Q::=*		*(processes)*
	$\texttt{in}\, n$	*enter n*		0	*inactivity*
	$\texttt{out}\, n$	*exit n*		$(\nu n)\ P$	*restriction*
	$\texttt{open}\, n$	*open n*		$P \mid Q$	*parallel composition*
				$!P$	*replication*
				$n[P]$	*ambient*
				$M.P$	*prefix*

Standard syntactical conventions are used: trailing zeros are omitted, and parallel composition has the least syntactic precedence. We refer to the usual notions of names, free names, and bound names of a process P, denoted by $n(P)$, $fn(P)$, $bn(P)$, respectively. We identify processes which are α-convertible, that is, can be made syntactically equal by a change of bound names. We adopt also the standard notation for substitutions: $P[m/n]$ denotes the process obtained by replacing in P any free occurrence of n with m (assuming the bound names of P are α-converted to avoid the conflicts with m). Similarly, $P\eta$ denotes the process obtained by applying the substitution $\eta : \mathcal{N} \to \mathcal{N}$.

The core of the semantics of MA consists of the reductions in Table 1 corresponding to the execution of capabilities. The semantics has also standard structural rules (Table 2) which use structural congruence to bring the participants of a potential interaction into contiguous positions (Table 3). The definition of $\equiv$ includes the standard rules for commuting the positions of parallel components, for stretching the scope of a restriction and for replication.

In the following we use $\rightarrow^*$ for the transitive and reflexive closure of $\rightarrow$. Moreover, we write $P \rightarrow_\equiv Q$ to say that either $P \rightarrow Q$ or $P \equiv Q$. Similarly for $P \rightarrow^*_\equiv Q$.

$n[\mathtt{in}\, m.\, P \mid Q] \mid m[R] \rightarrow m[n[P \mid Q] \mid R]$	(In)
$m[n[\mathtt{out}\, m.\, P \mid Q] \mid R] \rightarrow n[P \mid Q] \mid m[R]$	(Out)
$\mathtt{open}\, n.\, P \mid n[Q] \rightarrow P \mid Q$	(Open)

Table 1: Basic Reductions of Mobile Ambients

$P \rightarrow Q \Rightarrow (\nu n)\, P \rightarrow (\nu n)\, Q$	(Res)
$P \rightarrow Q \Rightarrow P \mid R \rightarrow Q \mid R$	(Par)
$P \rightarrow Q \Rightarrow n[P] \rightarrow n[Q]$	(Amb)
$(P' \rightarrow Q',\ P \equiv P',\ Q' \equiv Q) \Rightarrow P \rightarrow Q$	(Cong)

Table 2: Structural rules for Mobile Ambients

4 The Normal Semantics

The normal semantics aims at making easier the application of abstract interpretation, which is complicated by structural congruence (including α-conversion) and by the structural rules of the reduction semantics. The normal semantics is based on the intuitive representation of a process as a tree of ambients, each containing a set of active processes. We use a set, called a *topology*, to represent the tree of ambients, and a set, called a *configuration*, to represent the active processes contained in each ambient. For instance, the process

$$(\nu n)\ (n[\mathtt{in}\, k.\, P \mid \mathtt{out}\, k \mid m[\mathtt{out}\, n.\, Q]]) \mid k[!\mathtt{open}\, m] \tag{1}$$

is represented by the following topology and configuration (depicted also in Figure 1)

$$(\{\ n^{@},\ k^{@},\ m^{n}\},\{\ {}^{n}\mathtt{in}\, k.\, P,\ {}^{n}\mathtt{out}\, k,\ {}^{m}\mathtt{out}\, n.\, Q,\ {}^{k}!\mathtt{open}\, m\}).$$

$P \equiv P$	(Refl)
$Q \equiv P \Rightarrow P \equiv Q$	(Symm)
$P \equiv Q,\ Q \equiv R \Rightarrow P \equiv R$	(Trans)
$P \mid Q \equiv Q \mid P$	(Comm)
$(P \mid Q) \mid R \equiv P \mid (Q \mid R)$	(Ass)
$P \equiv Q \Rightarrow (\nu n)P \equiv (\nu n)Q$	(Res)
$P \equiv Q \Rightarrow P \mid R \equiv Q \mid R$	(Par)
$P \equiv Q \Rightarrow !P \equiv !Q$	(Bang)
$P \equiv Q \Rightarrow n[P] \equiv n[Q]$	(Amb)
$P \equiv Q \Rightarrow M.P \equiv M.Q$	(Pref)
$n \neq m \Rightarrow (\nu n)\ (\nu m)\ P \equiv (\nu m)\ (\nu n)\ P$	(Res-Com)
$n \notin fn(P) \Rightarrow (\nu n)\ (P \mid Q) \equiv P \mid (\nu n)\ Q$	(Res-Par)
$n \neq m \Rightarrow (\nu n)\ m[P] \equiv m[(\nu n)\ P]$	(Res-Amb)
$P \mid 0 \equiv P$	(Nil-Par)
$(\nu n)0 \equiv 0$	(Nil-Res)
$!P \equiv P \mid !P$	(Bang-Bang)

Table 3: Structural Congruence

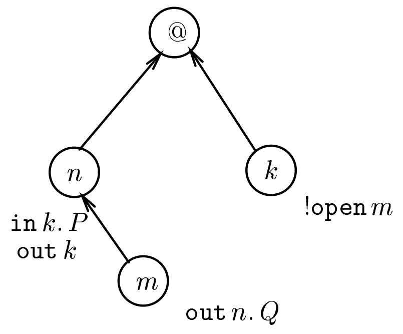

Figure 1: The representation of process (1)

The topology contains the pairs son-father: ${}_{m}{}^{n}$, because m is contained in n, ${}_{n}{}^{@}$ and ${}_{k}{}^{@}$, because n and k are contained in the outermost ambient that we call @. The configuration contains the processes executable inside any ambient: processes $\texttt{in}\, k.\, P$ and $\texttt{out}\, k$ inside n, process $\texttt{out}\, n.\, Q$ inside m, and process $\texttt{!open}\, m$ inside k.

The translation of a process into an equivalent pair of topology and configuration, as shown for process (1) above, presents two subtle problems. We need to: (i) distinguish two different occurrences of the same object in the process ; (ii) choose properly the names used for the removal of restrictions. In (1), for instance, we have eliminated the restriction operator by substituting n with a fresh name (in this case it suffices to take n itself).

To deal with these problems in a simple way we enhance the syntax of processes by properly attaching labels to capabilities, restrictions and ambients.

Provided that the labels assigned to capabilities, restrictions and ambients are distinct, we directly obtain a representation, where two copies of the same process or of an ambient with the same name are distinguished. For instance, consider the following labeled version of process $n[\texttt{in}\, m] \mid n[\texttt{in}\, k]$, where labels $\lambda, \mu, \gamma, \zeta$ are distinct one from each other

$$n_\lambda[\texttt{in}\, m_\gamma] \mid n_\mu[\texttt{in}\, k_\zeta]. \tag{2}$$

We obtain the following representation

$$(\{\, {}_{n_\lambda}{}^{@},\ {}_{n_\mu}{}^{@}\}, \{\, {}^{n_\lambda}\texttt{in}\, m_\gamma,\ {}^{n_\mu}\texttt{in}\, k_\zeta\})$$

where there are two copies of ambient n: one containing the capability $\texttt{in}\, m$ and the other one containing the capability $\texttt{in}\, k$.

We also use the labels attached to restrictions to find out the name, which is used to replace the bound name. To this aim, we adopt a special substitution function, which associates in a one to one fashion names to labels. Provided that all the labels are distinct and that the names associated to the labels of restrictions, do not appear in the process, the names introduced by the removal of restrictions are fresh. For instance, consider the following labeled process

$$(\nu n_\lambda)\ (n_\gamma[\texttt{in}\, m_\mu.\, P]) \mid (\nu m_\beta)\ m_\zeta[0] \tag{3}$$

where the labels $\lambda, \zeta, \gamma, \mu, \beta$ are distinct one from each other. Assume also that $\hat{n}$ and $\hat{m}$ are the distinct names associated to λ and β and that they do not appear in the process. We obtain the following representation

$$(\{\ {}_{\hat{n}_\gamma}{}^{@},\ {}_{\hat{m}_\zeta}{}^{@}\}, \{\ {}^{\hat{n}_\gamma}\mathtt{in}\, m_\mu . P\}).$$

The removal of the restrictions over n and m does not produce any conflict on names, as $\hat{m} \neq \hat{n}$, $\hat{m} \neq m$ and $\hat{n} \neq m$. The condition $\hat{m} \neq \hat{n}$ is implied by $\lambda \neq \beta$; the conditions $\hat{m} \neq m$ and $\hat{n} \neq m$ are ensured by the additional requirement concerning the names and the labels appearing in the process.

The requirements on labels and names explained above are formalised by the notion of well-labeled process (see Definition 4.2).

Labeled Processes. Let $\mathcal{L}$ be a set of *labels* (ranged over by ℓ, $\ell', \ldots$), and let $\mathcal{L}_I = \{\ell_i \mid \ell \in \mathcal{L}, i \in I\}$ be the corresponding set of *indexed labels* (ranged over by $\lambda, \mu, \gamma, \ldots$). Let $\widehat{\mathcal{N}}$ (ranged over by $\widehat{n}, \widehat{m}, \widehat{h}, \widehat{k}, \ldots$) be a set of names, such that $\mathcal{N} \cap \widehat{\mathcal{N}} = \emptyset$, and let $\widehat{\mathcal{N}}_I = \{\hat{n}_i \mid \hat{n} \in \widehat{\mathcal{N}}, i \in I\}$ be the corresponding set of *indexed names.*

We use the names $\widehat{\mathcal{N}}_I$ for the elimination of restrictions according to a substitution function $H_{\mathcal{L}_I}$ which assigns indexed names $\widehat{\mathcal{N}}_I$ to indexed labels $\mathcal{L}_I$. This is formalised by an injective function $H_{\mathcal{L}} : \mathcal{L} \to \widehat{\mathcal{N}}$ and by the corresponding injective function $H_{\mathcal{L}_I} : \mathcal{L}_I \to \widehat{\mathcal{N}}_I$, such that $H_{\mathcal{L}_I}(\ell_i) = \hat{n}_i$ if $H_{\mathcal{L}}(\ell) = \hat{n}$.

To have a more compact notation we may use when the distinction is not relevant: $n, m, h, \ldots$ to denote a generic element of $\widehat{\mathcal{N}}_I \cup \mathcal{N}$; $\hat{n}, \hat{m}, \hat{h}, \ldots$ to denote a generic element of $\widehat{\mathcal{N}}_I$.

Definition 4.1 (Labeled Processes) *The* labeled processes *are defined over names* $\mathcal{N} \cup \widehat{\mathcal{N}}_I$ *and indexed labels* $\mathcal{L}_I$ *according to the following syntax:*

M,N::=		*(capabilities)*	*P, Q::=*		*(processes)*
	$\mathtt{in}\, n$	*enter n*		0	*inactivity*
	$\mathtt{out}\, n$	*exit n*		$(\nu n_\lambda)\, P$	*restriction*
	$\mathtt{open}\, n$	*open n*		$P \mid Q$	*parallel composition*
				$!P$	*replication*
				$n_\lambda[P]$	*ambient*
				$M_\lambda . P$	*prefix*

We assume that all the notions presented in Section 3 are adapted in the obvious way to labeled processes. The definition of α-conversion only presents a subtle point: we require that the bound names can be changed but not their labels. We mean, for instance, that $(\nu n_\lambda)\, P$ is α-convertible to $(\nu k_\lambda)\, P[k/n]$, provided that $k \notin fn(P)$, and not to $(\nu k_\mu)\, P[k/n]$. In the following, we use $\Lambda(P)$ to denote the set of labels occurring in a labeled process P.

We introduce now the concept of well-labeled process, which formalises the requirements discussed for the processes (2) and (3) above. Conditions (i) and (ii) say that the labels are distinct and the names associated to the labels of restrictions are fresh names, meaning that they do not occur in the process. Example 4.10 shows, more in details, why these requirements are needed to translate a process into an equivalent representation.

Definition 4.2 (Well-labeled Processes) *A process P is* well-labeled *if: (i) for any $\lambda \in \Lambda(P)$, $H_{\mathcal{L}_I}(\lambda) \notin n(P)$; (ii) the (indexed) labels used in capabilities, ambients and restrictions are distinct one from each other.*

Over labeled processes we define a notion of equivalence, which is used in the definition of the collecting semantics (see Definition 4.8). A *renaming* of indexed labels is a function $\rho : \mathcal{L}_I \to \mathcal{L}_I$. The application of a renaming is denoted in the standard way by $P\rho$ and $P[\lambda/\mu]$. We denote by $dom(\rho)$ and $dom(\eta)$ the domains of a renaming ρ and a substitution η respectively. We also introduce a special class of renamings and substitutions:

- we say that $\rho_I : \mathcal{L}_I \to \mathcal{L}_I$ is a *re-indexing of labels* if, ρ_I is injective, and for any $\ell_i \in dom(\rho_I)$, we have $\rho_I(\ell_i) = \ell_j$;
- we say that $\eta_I : \hat{\mathcal{N}}_I \to \hat{\mathcal{N}}_I$ is a *re-indexing of names* if, η_I is injective and, for any $\hat{n}_i \in dom(\eta_I)$, we have $\eta_I(\hat{n}_i) = \hat{n}_j$.

We say that P and Q are *equivalent up to re-indexing* ($P \sim Q$) iff $P\rho_I\eta_I = Q$, for a re-indexing of labels ρ_I and a re-indexing of names η_I.

In the following, we use $\mathcal{A}$ (ranged over by a,b,c,...) for the set of labeled names n_λ, such that $n \in \mathcal{N} \cup \widehat{\mathcal{N}_I}$ and $\lambda \in \mathcal{L}_I$, augmented with the distinct symbol @ representing the outermost ambient. Furthermore, we say that a process P is *active* if $P = M_\lambda.Q$ or $P = !Q$. We use $\mathcal{P}$ and $\mathcal{AP}$ to denote the set of well-labeled processes (referred to as processes) and the subset of active well-labeled processes, respectively.

Remark 4.3 *It is worth mentioning that the labeling of processes is also exploited by the analyses of Sections 5 and 6. This approach is indeed typical of static techniques, in particular of Flow Logic [27]. The labeling of processes is used to gain precision, and also it allows the programmer to identify the exact piece of input syntax responsible for some detected security violation. The main difference here consists in the use of indexes both in labels $\mathcal{L}_I$ and in names $\hat{\mathcal{N}}_I$. The normal semantics and the second abstraction could have been defined also without introducing the indexes. Instead, the indexes are needed and fruitfully exploited by the first abstraction (see Example 5.10 and Example 5.12).*

States and Transitions. A state is a pair which consists of a topology and a configuration: the topology is a set of pairs, son-father, which form a tree, and the configuration is a set of pairs associating each active process to its enclosing ambient.

Definition 4.4 (States) *A* state *S is a pair (T, C) where*

1. *$T \in \wp((\mathcal{A} \setminus \{@\}) \times \mathcal{A})$ is a tree over a set of nodes $\mathcal{N}_S \subseteq \mathcal{A}$* [1]*;*
2. *$C \in \wp(\mathcal{A} \times \mathcal{AP})$ such that, for each $(a, P) \in C$, $a \in \mathcal{N}_S$.*

In a state (T, C) we call T a *topology* and C a *configuration.* The meaning of $(a, b) \in T$ (for short ${}_{a}^{b}$) is that a is a son of b. The meaning of $(a, P) \in C$ (for short ${}^{a}P$) is that P is an active process of ambient a.

We extend to states in the obvious way the notions of labels, renaming, substitution and equivalence up to re-indexing $\sim$. Since we are interested in states representing well-labeled processes we consider only well-labeled states. A state $S \in \mathcal{S}$ is *well-labeled* if: (i) for each $\lambda \in \Lambda(S)$, $H_{\mathcal{L}_I}(\lambda) \notin n(S)$; (ii) for any label $\lambda \in \Lambda(S)$ there is at most one object labeled by λ. In the following, we use $\mathcal{S}$ to denote the set of well-labeled states (referred to as states). Also, we assume $\subseteq$ and $\cup$ over states are defined component-wise.

In Table 4 we introduce the *normalisation function* $\delta : (\mathcal{A} \times \mathcal{P}) \rightarrow \mathcal{S}$ which is used to translate processes into states. Intuitively, $\delta(a, P)$ (for short $\delta\ {}^{a}P$) gives the state representing process P, assuming that P is contained in ambient a. We use δ both to derive the initial state from a process, and to handle the processes which become executable after a step. The initial state corresponding to a process P is therefore $\delta\ {}^{@}P$.

Rule **DRes** eliminates the restriction by replacing the bound name n with the name $H_{\mathcal{L}_I}(\lambda)$ associated to the indexed label λ. The definition of well-labeling ensures that $H_{\mathcal{L}_I}(\lambda)$ is a fresh name provided that P is a well-labeled process. Rule **DAmb** adds ambient b to the topology as son of the enclosing ambient a, and translates the process contained in b. Rule **DPar** gathers the processes and the topologies built in each of its two branches. Rules **DBang** and **DPref** simply add the active process to the configuration.

DRes	$\delta\ {}^{a}(\nu n_\lambda)\ P$	$=$	$\delta\ \ {}^{a}(P[H_{\mathcal{L}_I}(\lambda)/n])$
DAmb	$\delta\ {}^{a}b[P]$	$=$	$\delta\ {}^{b}P\ \cup\ (\{\ {}_{b}^{a}\}, \emptyset)$
DZero	$\delta\ {}^{a}0$	$=$	$(\emptyset,\ \emptyset)$
DPar	$\delta\ {}^{a}P \mid Q$	$=$	$\delta\ {}^{a}P\ \cup\ \delta\ {}^{a}Q$
DBang	$\delta\ {}^{a}!P$	$=$	$(\emptyset, \{{}^{a}!P\})$
DPref	$\delta\ {}^{a}M_\lambda.P$	$=$	$(\emptyset, \{{}^{a}M_\lambda.P\})$

Table 4: The normalisation function δ

The rules of Table 5 define the *transitions* between states. They realise the unfolding of replication, the movements in and out of ambients, and the opening of ambients. Due to the implicit representation of parallel composition, restriction and ambient in states,

[1]We refer to the standard definition of tree and root of a tree.

the standard structural rules and structural congruence of the reduction semantics are not needed. For notational convenience use the following abbreviation. We write $(T, C)\{\![a/b]\!\}$ to denote the state $(T[\,{}_{c}^{a}/\,{}_{c}^{b}], C[\,{}^{a}Q/\,{}^{b}Q])$ for any ambient c and process Q.

We comment the rules below. Rule **Bang** creates a fresh copy (equivalent up to re-indexing of labels) of the process under replication. To this aim, we use $new_{(T,C)}(P)$, which is defined as follows. Let $S \in \mathcal{S}$ be a state, we let $new_S(P) = P\rho_I$ where

- ρ_I is a re-indexing of labels such that $dom(\rho_I) = \Lambda(P)$;
- $P\rho_I$ is well-labeled;
- there is no $\lambda \in \Lambda(P\rho_I)$, such that either $\lambda \in \Lambda(S)$ or $H_{\mathcal{L}_I}(\lambda) \in n(S)$.

The definition of new_S ensures that $\delta\ {}^{a}new_S(P) \cup S$ is a well-labeled state, provided that S and P are well-labeled.

The last three rules correspond to the usual reduction rules of movements and opening (shown in Table 1). They use the normalisation function to handle the continuations. Rule **In** is applicable whenever there exists a parallel ambient named m. The rule modifies both the topology and the configuration according to the movement: (i) it updates the father of a, which is now m, (ii) it removes the executed capability, and it adds the continuation to the set of processes local to a. Rule **Out** acts in an analogous way. Rule **Open** modifies both the topology and the configuration according to the opening of ambient m: (i) it removes ambient m; (ii) it modifies the pointer to the father of any ambient and process which was within m. These processes and ambients are therefore acquired by ambient a when opening m.

Bang	$\dfrac{{}^{a}!P \in C}{(T, C) \mapsto \delta\ {}^{a}new_{(T,C)}(P) \ \cup\ (T, C)}$
In	$\dfrac{t =\ {}^{a}\mathtt{in}\, m_\gamma.P \in C \qquad {}_{a}^{b},\ {}_{m_\mu}{}^{b} \in T \qquad a \neq m_\mu}{(T, C) \mapsto \delta\ {}^{a}P \ \cup\ ((T \setminus \{\ {}_{a}^{b}\}) \cup \{\ {}_{a}^{m_\mu}\}, C \setminus \{t\})}$
Out	$\dfrac{t =\ {}^{a}\mathtt{out}\, m_\gamma.P \in C \qquad {}_{a}^{m_\mu},\ {}_{m_\mu}{}^{b} \in T \qquad a \neq m_\mu}{(T, C) \mapsto \delta\ {}^{a}P \ \cup\ ((T \setminus \{\ {}_{a}^{m_\mu}\} \cup \{\ {}_{a}^{b}\}, C \setminus \{t\})}$
Open	$\dfrac{t =\ {}^{a}\mathtt{open}\, m_\gamma.P \in C \qquad {}_{m_\mu}{}^{a} \in T \qquad a \neq m_\mu}{(T, C) \mapsto \delta\ {}^{a}P \ \cup\ ((T \setminus \{\ {}_{m_\mu}{}^{a}\}), (C \setminus \{t\}))\{\![a/m_\mu]\!\}}$

Table 5: Transitions $\mapsto$

The following theorem states the agreement between the transitions of Table 5 and the standard reduction semantics of Section 3. Let P be a well-labeled process. We denote by $\mathcal{E}(P)$ the process obtained by stripping off all the labels. We use $\mapsto^*$ for the transitive and reflexive closure of $\mapsto$.

We introduce also a condition on $a \in \mathcal{A}$ which guarantees that $\delta\ ^a P$ is a well-labeled state, provided that P is well-labeled (as formalised by Proposition A.13 of Appendix A). We first extend the notions of names $n(a)$ and labels $\Lambda(a)$. Hence, we let $n(a) = n$ and $\Lambda(a) = \lambda$, when $a = n_\lambda$, and we let $n(a) = \Lambda(a) = \emptyset$, when $a = @$.

We say that a is *fresh* for a labeled process P iff $\Lambda(a) \cap \Lambda(P) = \emptyset$, there is no $\mu \in \Lambda(a)$ such that $H_{\mathcal{L}_I}(\mu) \in (n(P) \cup n(a))$, and there is no $\mu \in \Lambda(P)$ such that $H_{\mathcal{L}_I}(\mu) \in n(a)$.

Theorem 4.5 (Equivalence) *Let P be a well-labeled process and let $a \in \mathcal{A}$ which is fresh for P.*

1. *If $\delta\ ^a P \mapsto S$, then there exist a well-labeled process Q, such that $\mathcal{E}(P) \rightarrow_\equiv \mathcal{E}(Q)$ and $\delta\ ^a Q = S$;*
2. *If $\mathcal{E}(P) \rightarrow Q$, then there exist a state S and a well-labeled process Q', such that $\delta\ ^a P \mapsto^* S$, $\delta\ ^a Q' = S$ and $Q \equiv \mathcal{E}(Q')$.*

The proof of Theorem 4.5 is rather complex and is shown in the Appendix A.

Corollary 4.6 *Let P be a well-labeled process.*

1. *If $\delta\ ^@ P \mapsto S$, then there exist a well-labeled process Q, such that $\mathcal{E}(P) \rightarrow_\equiv \mathcal{E}(Q)$ and $\delta\ ^@ Q = S$;*
2. *If $\mathcal{E}(P) \rightarrow Q$, then there exist a state S and a well-labeled process Q', such that $\delta\ ^@ P \mapsto^* S$, $\delta\ ^@ Q' = S$ and $Q \equiv \mathcal{E}(Q')$.*

Proof: From Theorem 4.5 using the fact that @ is fresh for any well-labeled process P. $\square$

The result can be extended straightforwardly to weak reductions.

Corollary 4.7 *Let P be a well-labeled process.*

1. *If $\delta\ ^@ P \mapsto^* S$, then there exist a well-labeled process Q, such that $\mathcal{E}(P) \rightarrow^*_\equiv \mathcal{E}(Q)$ and $\delta\ ^@ Q = S$;*
2. *If $\mathcal{E}(P) \rightarrow^* Q$, then there exist a state S and a well-labeled process Q', such that $\delta\ ^@ P \mapsto^* S$, $\delta\ ^@ Q' = S$ and $Q \equiv \mathcal{E}(Q')$.*

The collecting semantics. We define the core of the abstract interpretation framework, the *collecting semantics*. The domain is the power-set of (well-labeled) states up to re-indexing. We use $[S]$ to denote the equivalence class of a state S with respect to $\sim$, and we use $\mathcal{S}_{/\sim}$ to denote the corresponding quotient set. For readability, we use $\subseteq$ and $\cup$ for $\subseteq_{/\sim}$ and $\cup_{/\sim}$.

Definition 4.8 *Let* $\mathcal{S}^\natural = \wp(\mathcal{S}_{/\sim})$. *The* concrete domain *is* $\langle \mathcal{S}^\natural, \subseteq \rangle$.

The concrete semantics is defined in a standard way as the least fixed-point of a function, which collects all the states reachable from the initial state.

Definition 4.9 (Collecting Semantics) *Let* $S_2 \in \mathcal{S}, S^\natural \in \mathcal{S}^\natural$ *and let* P *be a well-labeled process. We define* $\mathfrak{S}_{Coll}[\![P]\!] =$ *lfp* $F(\delta\ {}^{@}P)$ *for the function* $F : \mathcal{S} \to (\mathcal{S}^\natural \to \mathcal{S}^\natural)$ *such that* $F(S_2) = \Psi_{S_2}$ *and*

$$\Psi_{S_2}(S^\natural) = \{[S_2]\} \cup \bigcup_{S \in \{S_3 | S_1 \mapsto S_3,\ [S_1] \in S^\natural\}} \{[S]\}.$$

Examples. We start discussing the normalisation function δ, and we explain why this is correct (in the sense of Theorem 4.5) only when applied to well-labeled processes.

Example 4.10 *The condition (ii) of Definition 4.2 ensures that two occurrences of the same object are distinguished. Consider the not well-labeled version of process* $n[\mathtt{in}\, m] \mid n[\mathtt{in}\, k]$,

$$P = n_\lambda[\mathtt{in}\, m_\gamma] \mid n_\lambda[\mathtt{in}\, k_\zeta].$$

We obtain the following representation

$$\delta\ {}^{@}P = (\{\ {}_{n_\lambda}{}^{@}\}, \{\ {}^{n_\lambda}\mathtt{in}\, m_\gamma,\ {}^{n_\lambda}\mathtt{in}\, k_\zeta\}).$$

This representation differs significantly from that shown at the beginning of the section for the well-labeled process (2). In fact there is only one ambient n *which contains both* $\mathtt{in}\, m$ *and* $\mathtt{in}\, k$. *This representation is obviously not correct as ambient* n *may interact both with* m *and with* k.

The condition (i) of Definition 4.2 concerns the relation between the names in $\widehat{\mathcal{N}}_I$ *and the labels* $\mathcal{L}_I$, *and ensures precisely that there is no clash of names when the restrictions are removed. Consider the following not well-labeled process*

$$Q = (\nu n_\lambda)\ (n_\gamma[\mathtt{in}\, \hat{m}_\mu.\, P]) \mid (\nu m_\beta)\ m_\zeta[0]$$

where $H_{\mathcal{L}_I}(\lambda) = \hat{n}$ *and* $H_{\mathcal{L}_I}(\beta) = \hat{m}$. *We obtain the following representation*

$$\delta\ {}^{@}Q = (\{\ {}_{\hat{n}_\gamma}{}^{@},\ {}_{\hat{m}_\zeta}{}^{@}\}, \{\ {}^{\hat{n}_\gamma}\mathtt{in}\, \hat{m}_\mu.\, P\}).$$

This representation is not correct, differently from the one obtained for the well-labeled process (3) shown at the beginning of the section. The bound name $\hat{m}$ *is known to the process contained inside* $\hat{n}$, *and consequently* $\hat{n}$ *can move inside* $\hat{m}$.

We give some examples to illustrate the normal semantics. To simplify the presentation in the collecting semantics states stand for their equivalence classes up to re-indexing. The following example shows an ambient n, which moves inside an ambient k, and there is opened unleashing no capability of movement inside k.

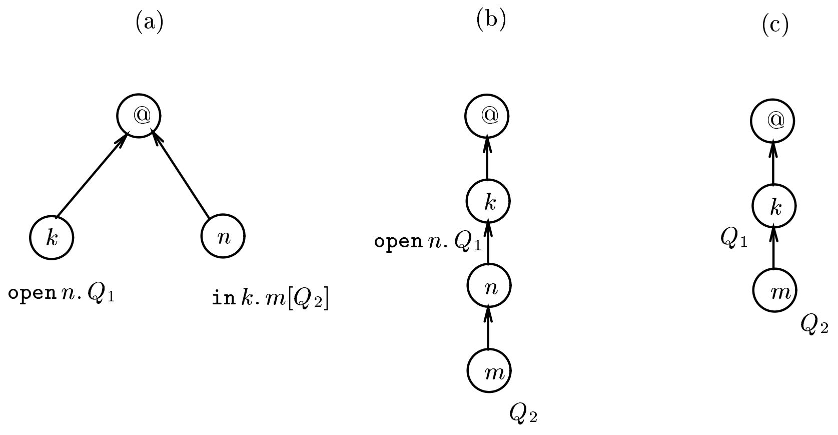

Figure 2: Some transitions of process P

Example 4.11 *Consider the (well-labeled) process*

$$P = n_\lambda[\mathtt{in}\, k_\epsilon.\, m_\zeta[Q_2]] \mid k_\mu[\mathtt{open}\, n_\beta.\, Q_1]$$

Figure 2 shows some states, which are reachable from the initial state representing process P which is state (a)[2]*. State (b) is obtained from state (a) by applying rule* **In**. *This shows that ambient n moves inside k carrying any capability and ambient it contains. State (c) is obtained from state (b) by applying rule* **Open**; *ambient n, when opened inside k, unleashes ambient m which has as a local process Q_2.*

By assuming that $Q_1 = Q_2 = 0$, the collecting semantics of P contains only states (a),(b) and (c) of Figure 2. We have $\mathfrak{S}_{Coll}[\![P]\!] = \{S_0, S_1, S_2\}$ such that

$$S_0 = (\{\ {n_\lambda}^{@},\ {k_\mu}^{@}\}, \{\ {}^{n_\lambda}\mathtt{in}\, k_\epsilon.\, m_\zeta[0],\ {}^{k_\mu}\mathtt{open}\, n_\beta\})$$

$$S_1 = (\{\ {n_\lambda}^{k_\mu},\ {m_\zeta}^{n_\lambda},\ {k_\mu}^{@}\}, \{\ {}^{k_\mu}\mathtt{open}\, n_\beta\})$$

$$S_2 = (\{\ {m_\zeta}^{k_\mu},\ {k_\mu}^{@}\}, \emptyset).$$

The following example stresses an important aspect concerning indexes and replication: the unfolding of replication produces (by means of *new*) processes which are equivalent up to re-indexing of labels. The link between the processes produced by replication expressed by the indexes is suitably exploited by the first abstraction (see Examples 5.10 and 5.12). Also, the example explains better the technique used to remove the restriction operator and its interplay with replication.

[2]We have omitted the labels to simplify the picture.

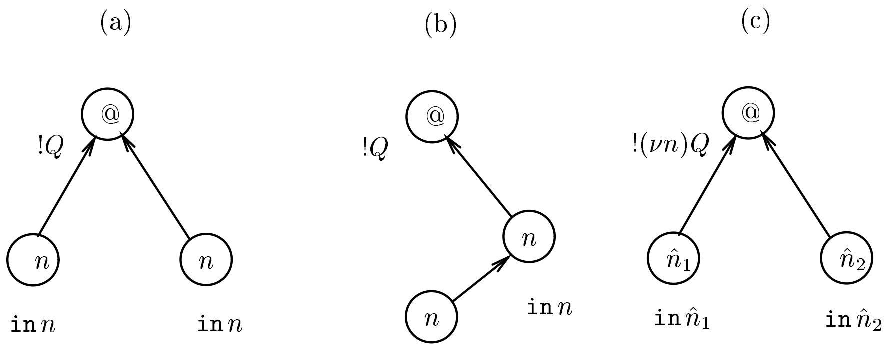

Figure 3: An example of replication

Example 4.12 *Consider the well-labeled process $Q = n_\lambda[\mathtt{in}\, n_\gamma]$, where $\lambda = \ell_1$ and $\gamma = \ell'_1$. The initial state modeling process $!Q$ is $(\emptyset,\ {}^{@}!Q)$. Every unfolding of replication is modeled by the addition of $\delta\ {}^{@}(Q\rho_j)$ (see rule* **Bang***), where*

$$Q\rho_j = n_{\ell_j}[\mathtt{in}\, n_{\ell'_j}]$$

for a new index j. Hence, a new ambient n_{ℓ_j} is added representing a new copy of ambient n. For instance, after two applications of rule **Bang** *the state (a) depicted in Figure 3 is reached*[3]*. Any ambient n_{ℓ_j} may enter inside any other n_{ℓ_h} provided that $h \neq j$. For instance, by applying rule* **In** *state (b) of Figure 3 is obtained.*

Consider instead the well-labeled process $(\nu n_\mu)\ Q$, where the name n is restricted and $\mu = \ell''_1$ such that $H_{\mathcal{L}}(\ell'') = \hat{n}$. The initial state modeling process $!(\nu n_\mu)\ Q$ is $(\emptyset,\ {}^{@}!(\nu n_\mu)\ Q)$. Every unfolding of replication is modeled by the addition of $\delta\ {}^{@}((\nu n_\mu)\ Q)\rho_j$, where

$$((\nu n_\mu)\ Q)\rho_j = (\nu n_{\ell''_j})\ n_{\ell_j}[\mathtt{in}\, n_{\ell'_j}]$$

for a new index j. Function $H_{\mathcal{L}_I}$ assigns to any label ℓ''_j the new name $\hat{n}_j$ which is used to substitute n. Hence, a new ambient $(\hat{n}_j)_{\ell_j}$ is added with a new name $\hat{n}_j$. For instance, after two applications of rule **Bang** *the state (c) depicted in Figure 3 is reached from the initial state. Since the names $\hat{n}_1, \hat{n}_2$ are distinct, then the ambients cannot in this case interact with each other.*

The difference between $!Q$ and $!(\nu n_\mu)\ Q$ is reflected by their collecting semantics shown below. We have $\mathfrak{S}_{Coll}[\![!Q]\!] = \bigcup_{j\in\{0,\dots,\infty\}} X_j$ where

- *$X_0 = \{(\emptyset,\ {}^{@}!Q)\}$;*
- *X_j is the minimal set of states $S = (T, C)$, such that $fn(S) = \{n\}$, and $\Lambda(S) = \bigcup_{i\in\{1,\dots,j\}}\{\ell_i, \ell'_i\}$ and ${}^{@}!Q \in C$. Moreover, for each $i \in \{1,\dots,j\}$ either ${}_{n_{\ell_i}}{}^{@} \in T$ and ${}^{n_{\ell_i}}\mathtt{in}\, n_{\ell'_i} \in C$, or ${}_{n_{\ell_i}}{}^{n_{\ell_h}} \in T$, with $h \neq i$, and ${}^{n_{\ell_i}}\mathtt{in}\, n_{\ell'_i} \notin C$.*

[3]As usual we have omitted labels to simplify the picture.

We have $\mathfrak{S}_{Coll}[\![!(\nu n_\mu)\ Q]\!] = \bigcup_{j\in\{0,\infty\}} S_j$ *where*

$$S_0 = \quad (\emptyset,\ {}^{@}!(\nu n_\mu)\,Q)$$

$$S_j = \quad (\bigcup_{i\in\{1,\dots,j\}}\ {}_{(\hat{n}_i)_{\ell_i}}^{@}, \bigcup_{i\in\{1,\dots,j\}}\ {}^{(\hat{n}_i)_{\ell_i}}\mathtt{in}\,(\hat{n}_i)_{\ell'_i} \cup \ \{\ {}^{@}!(\nu n_\mu)\,Q\}).$$

5 A First Abstraction

We devise a first abstraction aimed at capturing the following property about all the states reachable from the initial state representing a process P: for each ambient n, which ambients and capabilities can be contained (at top level) inside n, when n is within an ambient h. This is formalised by an abstraction, which merges a set of states into a unique abstract state, and modifies the topologies and the configurations according to the following ideas.

- We add to each pair of the topology and of the configuration an additional information which refers to the father of the enclosing ambient.

  Consider for instance the states

$$S_1 = (\{\ {}_{a}^{@},\ {}_{b}^{@}\},\ {}^{a}\mathtt{in}\,k_\mu.\,\mathtt{in}\,m_\gamma) \tag{4}$$
$$S_2 = (\{\ {}_{a}^{b},\ {}_{b}^{@}\},\ {}^{a}\mathtt{in}\,m_\gamma). \tag{5}$$

  They are represented by the following abstract states, respectively

$$S_1^\diamond = (\{\ {}_{b}^{@^\top},\ {}_{a}^{@^\top}\},\{\ \ {}^{a^{@}}\mathtt{in}\,k_\mu.\,\mathtt{in}\,m_\gamma\})$$

$$S_2^\diamond = (\{\ {}_{a}^{b^{@}},\ {}_{b}^{@^\top}\},\{\ \ {}^{a^{b}}\mathtt{in}\,m_\gamma\}).$$

  In $S_1^\diamond$ we have $\ {}^{a^{@}}\mathtt{in}\,k_\mu.\,\mathtt{in}\,m_\gamma$ as $\mathtt{in}\,k_\mu.\,\mathtt{in}\,m_\gamma$ is an active process inside ambient a, when a is within @. The same happens in the topology. For instance ${}_{a}^{@^\top}$ says that ambient a is a son of the top level ambient @, when @ is within $\top$ [4]. The abstract state $S_2^\diamond$ is obtained similarly.

  To understand the relevance of the information we have introduced, it is necessary to look at the abstraction of the set of states $\{S_1, S_2\}$. This is given by the union of $S_1^\diamond$ and $S_2^\diamond$ (depicted also in Figure 4) [5]

$$S^\diamond = (\{\ {}_{a}^{b^{@}},\ {}_{b}^{@^\top},\ {}_{a}^{@^\top}\},\{\ \ {}^{a^{b}}\mathtt{in}\,m_\gamma,\ \ {}^{a^{@}}\mathtt{in}\,k_\mu.\,\mathtt{in}\,m_\gamma\}).$$

[4]The extra symbol $\top$ is used to model the ambient enclosing @ and is mentioned for uniformity.

[5]Rounded arrows represent the partial topology, pointing from an object to the link representing the relevant pair son/father. As usual we have omitted labels.

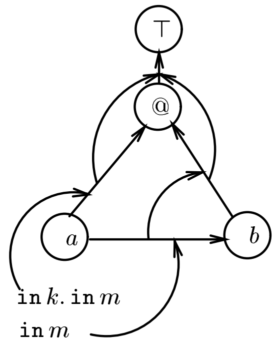

Figure 4: The abstract state $S^\diamond$

The abstract version of $\{S_1, S_2\}$ shows that the abstract topologies, differently from the concrete topologies, may not form a tree. For instance, in $S^\diamond$ ambient a has two fathers, namely ambients b and @. The additional information permits to distinguish between the multiple fathers of ambient a, and consequently to argue that the processes and the ambients contained inside a may depend on where a is located. For instance, in $S^\diamond$ we have that: when a is within @ process $\mathtt{in}\,k_\mu.\,\mathtt{in}\,m_\gamma$ is executable inside a; when a is within b instead process $\mathtt{in}\,m_\gamma$ is executable inside a.

We call this additional information, the *partial topology*, as it gives us a partial view of the shape of the tree-like structure (the topology) of the state, which contains the pair of ambients, son-father, or the pair associating each active process to its enclosing ambient.

- We abstract indexes by keeping only the following information: whether there is *at most* one occurrence or *many* occurrences of an object.

  Consider for instance the following states

$$S_1 = (\ n_{\ell'_1}{}^{@},\ {}^{n_{\ell'_1}}\mathtt{open}\,m_{\ell_1}) \qquad (6)$$

$$S_2 = (\ n_{\ell'_2}{}^{@}, \{\ {}^{n_{\ell'_2}}\mathtt{open}\,m_{\ell_1},\ {}^{n_{\ell'_2}}\mathtt{open}\,m_{\ell_2}\}). \qquad (7)$$

  They are represented by the following abstract states $S_1^\diamond$ and $S_2^\diamond$, respectively

$$S_1^\diamond = (\ n_{\ell'_1}{}^{@^\top},\ \ {}^{n_{\ell'_1}{}^{@}}\ \mathtt{open}\,m_{\ell_1})$$

$$S_2^\diamond = (\ n_{\ell'_1}{}^{@^\top},\ \ {}^{n_{\ell'_1}{}^{@}}\ \mathtt{open}\,m_{\ell_\omega}).$$

  The capability $\mathtt{open}\,m$ in state S_1 is represented by $\mathtt{open}\,m_{\ell_1}$, and the two copies of $\mathtt{open}\,m$ in state S_2 are represented by $\mathtt{open}\,m_{\ell_\omega}$. The label ℓ_1 has multiplicity one, and shows that there is one occurrence of the corresponding object; the label ℓ_ω has multiplicity ω, and shows that there are many occurrences of the corresponding object equivalent up to re-indexing.

By abstracting the set of states $\{S_1, S_2\}$ we obtain the following abstract state

$$S^\diamond = (\ {n_{\ell'_\omega}}^{@^\top},\ \ {}^{n_{\ell'_\omega}}{}^{@}\, \mathtt{open}\, m_{\ell_\omega}).$$

In the abstract state $S^\diamond$ both labels have multiplicity ω showing that there are many occurrences of ambient n and of capability $\mathtt{open}\, m$. The abstract state $S^\diamond$ is obtained by taking the least upper bound of $S_1^\diamond$ and $S_1^\diamond$ with respect to a particular ordering over abstract states which realises the union and modifies the multiplicity of objects accordingly.

The abstraction of indexes explained above is needed to achieve a computable analysis, in that we may have infinite processes equivalent up to re-indexing (see Example 4.12).

Abstract domain. Let $\mathcal{L}^\diamond = \{\ell_1, \ell_\omega \mid \ \ell \in \mathcal{L}\}$ (ranged over by $\lambda^\diamond, \mu^\diamond, \gamma^\diamond, \ldots$) be the set of *abstract labels*, and let $\mathcal{N} \cup \widehat{\mathcal{N}}$ (ranged over by $n^\diamond, m^\diamond, k^\diamond, h^\diamond, \ldots$) be the set of *abstract names*. Let $\mathcal{A}^\diamond$ (ranged over by $a^\diamond, b^\diamond, c^\diamond, \ldots$) be the set of *abstract labeled names* $n^\diamond{}_{\lambda^\diamond}$, augmented with the symbols @ and $\top$. The relation between names and labels is modified accordingly. We define $H_{\mathcal{L}^\diamond} : \mathcal{L}^\diamond \to \widehat{\mathcal{N}}$ such that $H_{\mathcal{L}^\diamond}(\ell_1) = H_{\mathcal{L}^\diamond}(\ell_\omega) = H_{\mathcal{L}}(\ell)$.

The abstract labeled processes are built according to the syntax of Definition 4.1 over names $\mathcal{N} \cup \widehat{\mathcal{N}}$ and labels $\mathcal{L}^\diamond$. We assume that all the previously defined notions on processes are adapted to abstract processes in the expected way. As in the concrete case we consider only well-labeled processes.

Definition 5.1 (well-labeled) *An abstract process $P^\diamond$ is* well-labeled *if : (i) $\ell_1 \in \Lambda(P^\diamond)$ implies $\ell_\omega \notin \Lambda(P^\diamond)$; (ii) for any label $\lambda \in \mathcal{L}^\diamond$, such that $\lambda = \ell_1$, there is at most one object labeled by λ.*

In the following we use $\mathcal{P}^\diamond$ and $\mathcal{AP}^\diamond$ to denote the set of well-labeled abstract processes (referred to as abstract processes) and active well-labeled abstract processes, respectively.

Definition 5.2 (Abstract States) *An* abstract state *$S^\diamond$ is a pair $(T^\diamond, C^\diamond)$ where*

1. *$T^\diamond \in \wp((\mathcal{A}^\diamond \setminus \{@, \top\}) \times (\mathcal{A}^\diamond \setminus \{\top\}) \times \mathcal{A}^\diamond)$;*
2. *$C^\diamond \in \wp(((\mathcal{A}^\diamond \setminus \{\top\}) \times \mathcal{A}^\diamond) \times \mathcal{AP}^\diamond)$.*

In an abstract state $S^\diamond = (T^\diamond, C^\diamond)$ we call $T^\diamond$ the topology and $C^\diamond$ the configuration. The meaning of $(a^\diamond, b^\diamond, c^\diamond) \in T^\diamond$ (for short ${a^\diamond}^{{b^\diamond}^{c^\diamond}}$) is that ambient $a^\diamond$ is a son of ambient

$b^\diamond$, when $b^\diamond$ is within $c^\diamond$. The meaning of $((a^\diamond, b^\diamond), P^\diamond) \in C^\diamond$ (for short ${}^{a^\diamond b^\diamond} P^\diamond$) is that $P^\diamond$ is executable inside ambient $a^\diamond$, when $a^\diamond$ is within $b^\diamond$.

We assume that all the previously defined notions on states are adapted to abstract states in the expected way. As in the concrete case we consider only well-labeled states. An abstract state $S^\diamond = (C^\diamond, T^\diamond)$ is *well-labeled* if conditions (i) and (ii) of Definition 5.1 hold (with $P^\diamond$ replaced by $S^\diamond$). We use $\mathcal{S}^\diamond$ to denote the set of well-labeled abstract states (referred to as abstract states).

We now introduce a proper ordering over abstract states [6].

Definition 5.3 *We define $\subseteq^\diamond$ as the minimal ordering over $\mathcal{S}^\diamond$, such that $S^\diamond \subseteq S'^\diamond$ implies $S^\diamond \subseteq^\diamond S'^\diamond$, and such that $S^\diamond \subseteq^\diamond S^\diamond[\ell_\omega/\ell_1]$. We use $\cup^\diamond$ for the least upper bound with respect to $\subseteq^\diamond$.*

The ordering reflects the intuition that ℓ_1 is more precise than ℓ_ω. For instance, assuming that $\lambda = \ell_1$ and $\gamma = \ell_\omega$, we have

$$({}_{n_\lambda}{}^{b^\diamond},\ {}^{n_\lambda b^\diamond} P^\diamond) \cup^\diamond ({}_{n_\gamma}{}^{b^\diamond},\ {}^{n_\gamma b^\diamond} P^\diamond) = ({}_{n_\gamma}{}^{b^\diamond},\ {}^{n_\gamma b^\diamond} P^\diamond).$$

Definition 5.4 *The* abstract domain *is $\langle \mathcal{S}^\diamond, \subseteq^\diamond \rangle$.*

To simplify the presentation in the following we may omit the over-script $-^\diamond$ for any syntactic category, when the meaning is clear from the context.

The Galois connection. We present now the relation between the concrete and the abstract domain establishing a Galois connection (see Definition 2.1). We formalise the ideas explained at the beginning of the section. A single state is abstracted

1. by introducing the partial topology both in the topology and in the configuration;
2. by replacing the indexed labels $\mathcal{L}_I$ with the abstract labels $\mathcal{L}^\diamond$, and by substituting the names $\widehat{\mathcal{N}}_I$ with the abstract names $\widehat{\mathcal{N}}$.

To remove the indexes according to 2., we introduce a special renaming, that depend on the state which is abstracted, and a special substitution. Let $S \in \mathcal{S}$ be a state. We define a renaming $\rho^\diamond_S : \mathcal{L}_I \to \mathcal{L}^\diamond$ such that $\rho^\diamond_S(\ell_i) = \ell_\omega$, if there exist j with $i \neq j$ such that $\ell_i, \ell_j \in \Lambda(S)$, and $\rho^\diamond_S(\ell_i) = \ell_1$ otherwise. Moreover, we define a substitution $\eta^\diamond : \widehat{\mathcal{N}}_I \to \widehat{\mathcal{N}}$ such that $\eta^\diamond(\hat{n}_i) = \hat{n}$.

A set of states is abstracted by taking the least upper bound with respect to $\subseteq^\diamond$ of the abstraction of each element.

Definition 5.5 *Let $S^\natural \in \mathcal{S}^\natural$, $(T, C) \in \mathcal{S}$ and $S^\diamond \in \mathcal{S}^\diamond$. We define $\alpha^\diamond : \mathcal{S}^\natural \to \mathcal{S}^\diamond$ and $\gamma^\diamond : \mathcal{S}^\diamond \to \mathcal{S}^\natural$ as follows*

[6] As usual we assume that $\subset$ and $\cup$ are defined component-wise.

1. $\alpha^\diamond((T,C)) = (T^\diamond, C^\diamond)\rho^\diamond_{(T,C)}\eta^\diamond$, *where* [7]

$$T^\diamond = \{\ {}_a^{bc} \mid {}_a^{b},\ {}_b^{c} \in T\}$$

$$C^\diamond = \{\ {}_a^{\ b}P \mid {}_a^{b} \in T,\ {}^aP \in C\};$$

2. $\alpha^\diamond(S^\natural) = \bigcup^\diamond_{[S]\in S^\natural}\alpha^\diamond([S])$, *where* $\alpha^\diamond([S]) = \bigcup^\diamond_{S'\in[S]}\alpha^\diamond(S')$;

3. $\gamma^\diamond(S^\diamond) = \bigcup_{[S]\in\{[S'] \mid \alpha^\diamond(\{[S']\})\subseteq^\diamond S^\diamond\}}\{[S]\}$.

Note that in the definition above (case 1.) we have introduced an auxiliary abstraction function $\alpha^\diamond : \mathcal{S} \to \mathcal{S}^\diamond$ which maps a state into an abstract state. This is used to define the abstraction function $\alpha^\diamond : \mathcal{S}^\natural \to \mathcal{S}^\diamond$ which maps a set of states up to re-indexing into an abstract state (case 2.).
The pair of functions defined above is a Galois connection.

Theorem 5.6 *The pair of functions* $(\alpha^\diamond, \gamma^\diamond)$ *is a Galois connection between* $\langle \mathcal{S}^\natural, \subseteq\rangle$ *and* $\langle \mathcal{S}^\diamond, \subseteq^\diamond\rangle$.

The proof of Theorem 5.6 is shown in the Appendix B.1.

Abstract semantics. The abstract semantics is defined by an abstract normalisation function and by abstract transitions, which adapt the normalisation function of Table 4 and the transitions of Table 5 to the abstract domain.

The abstract normalisation function $\delta^\diamond : (A^\diamond \times \mathcal{A}^\diamond) \times \mathcal{P}^\diamond \to \mathcal{S}^\diamond$ is defined in Table 6 (as usual $\delta^\diamond\ {}_a^{\ b}P$ stands for $\delta^\diamond((a,b),P)$). The main differences with respect to δ are that $\delta^\diamond$ deals with the partial topology and with the multiplicity. For instance, rule **DAmb**$^\diamond$ adds ${}_c^{ab}$ to the topology instead of ${}_c^{a}$. Similarly, rule **DPref**$^\diamond$ adds ${}_a^{\ b}M_\lambda.P$ instead of ${}^aM_\lambda.P$. Also the rules use $\cup^\diamond$ in place of $\cup$ to properly handle labels with multiplicity.

The transition rules are shown in Table 7. For notational convenience we use the following abbreviations. We write $(T,C)\{\![a^d/b^c]\!\}$ to denote the abstract state $(T[e^{a^d}/e^{b^c}]$, $C[\ {}_a^{\ d}Q/\ {}_b^{\ c}Q])$ for any ambient e and process Q. Also we use

$$C \setminus^\diamond\{\ {}_a^{\ b}M_{\lambda^\diamond}.P\} = \begin{cases} C \text{ if } \lambda^\diamond = \ell_\omega \\ \\ (C \setminus\ {}_a^{\ b}M_{\lambda^\diamond}.P) \quad \text{if } \lambda^\diamond = \ell_1 \end{cases}$$

The rules are rather complex, it is worth explaining the most interesting cases to point out especially the role of the partial topology and of the multiplicity. Notice that, in each rule, the abstract normalisation function $\delta^\diamond$ is used in place of δ to handle the continuations.

[7]We are assuming that the symbols @ and $\top$ are introduced, when needed, to give a father and grandfather to any ambient. In particular, using $\top$ for the father of @, and @ for the father of the root, when different from @.

DRes$^\diamond$	$\delta^\diamond\ {}_{a}^{b}(\nu n_\lambda)\, P$	$=$	$\delta^\diamond\ {}_{a}^{b}(P[H_{\mathcal{L}^\diamond}(\lambda)/n])$
DAmb$^\diamond$	$\delta^\diamond\ {}_{a}^{b}c[P]$	$=$	$\delta^\diamond\ {}_{c}^{a}P \cup^\diamond(\{\ {}_{c}^{a^{b}}\}, \emptyset)$
DZero$^\diamond$	$\delta^\diamond\ {}_{a}^{b}0$	$=$	$(\emptyset,\ \emptyset)$
DPar$^\diamond$	$\delta^\diamond\ {}_{a}^{b}P \mid Q$	$=$	$\delta^\diamond\ {}_{a}^{b}P \cup^\diamond \delta^\diamond\ {}_{a}^{b}Q$
DBang$^\diamond$	$\delta^\diamond\ {}_{a}^{b}!P$	$=$	$(\emptyset, \{\ {}_{a}^{b}!P\})$
DPref$^\diamond$	$\delta^\diamond\ {}_{a}^{b}M_\lambda.\, P$	$=$	$(\emptyset, \{\ {}_{a}^{b}M_\lambda.\, P\})$

Table 6: The normalisation function $\delta^\diamond$

Bang$^\diamond$ The rule unfolds replication by creating a copy of the process, where every label has multiplicity ω, instead of creating a fresh copy (equivalent up to re-indexing). We use new_ω, which is defined as $new_\omega(P) = P\rho$ for the renaming ρ, where $\rho(\ell_1) = \ell_\omega$ for any $\ell_1 \in \Lambda(P)$.

In$^\diamond$ The rule is applicable whenever there exists an ambient named m, which is contained in the father b of a, when in both cases b is within c. The multiplicity of ambient m influences the movement, meaning that m can move inside itself only if its multiplicity is ω (see the side-condition of the $(a = m_{\ell'_1} \Rightarrow \ell'_1 \neq \mu)$).

The movement is realised by a modification both of the topology and of the configuration: (i) ${}_{a}^{m^{b}}$ is added to the topology; (ii) the continuation P and the processes, which are active inside a in parallel with $\mathtt{in}\, m.\, P$, are added to the set of processes active, when a is within m (similarly for the ambients contained inside a); (iii) the process $\mathtt{in}\, m.\, P$ is added to the set of processes executable, when a is within m, depending on the multiplicity of the capability $\mathtt{in}\, m$. In particular, it is not added when $\mathtt{in}\, m$ has multiplicity one, as it has been consumed, after a has moved inside m (this is why we consider $C\setminus^\diamond\{t\}$).

Open$^\diamond$ The rule is applicable whenever there exists an ambient named m contained in a, when a is inside b. The effect of the opening of m inside a is that, all the processes and ambients, which are contained in m, when m is inside a, are acquired by a. The partial topology is used to determine precisely those processes and ambients.

The abstract semantics is defined as follows.

Definition 5.7 (The abstract semantics) *Let $S_1^\diamond, S_2^\diamond \in \mathcal{S}^\diamond$, and let P be a well-labeled process. We define $\mathfrak{S}_{Coll^\diamond}[\![P]\!] =$ lfp $F^\diamond(\alpha^\diamond(\delta\ {}^{@}P)\ )$, for the function $F^\diamond : \mathcal{S}^\diamond \to (\mathcal{S}^\diamond \to \mathcal{S}^\diamond)$ such that $F^\diamond(S_2^\diamond) = \Psi^\diamond_{S_2^\diamond}$ and*

$$\Psi^\diamond_{S_2^\diamond}(S_1^\diamond) = S_2^\diamond \cup^\diamond \bigcup\nolimits^\diamond_{S^\diamond \in \{S_3^\diamond \mid S_1^\diamond \mapsto^\diamond S_3^\diamond\}} S^\diamond.$$

$$\mathbf{Bang}^{\diamond} \quad \frac{{}_{a}^{b}!P \in C}{(T,C) \mapsto^{\diamond} \delta^{\diamond}\ {}_{a}^{b}new_{\omega}(P) \cup^{\diamond} (T,C)}$$

$$\mathbf{In}^{\diamond} \quad \frac{t = {}_{a}^{b}\mathtt{in}\, m_{\lambda}.P \in C \qquad {}_{a}^{bc},\ {}_{m_{\mu}}^{bc} \in T \qquad (a = m_{\ell'_1} \Rightarrow \ell'_1 \neq \mu)}{(T,C) \mapsto^{\diamond} \delta^{\diamond}\ {}_{a}^{m_{\mu}}P \cup^{\diamond} (T,C) \cup^{\diamond} (T \cup^{\diamond} \{ {}_{a}^{m_{\mu}^{b}} \}, C \setminus^{\diamond} \{t\})\{\![a^{m_{\mu}}/a^{b}]\!\}}$$

$$\mathbf{Out}^{\diamond} \quad \frac{t = {}_{a}^{m_{\mu}}\mathtt{out}\, m_{\lambda}.P \in C \qquad {}_{a}^{m_{\mu}^{c}},\ {}_{m_{\mu}}^{cb} \in T \qquad (a = m_{\ell'_1} \Rightarrow \ell'_1 \neq \mu)}{(T,C) \mapsto^{\diamond} \delta^{\diamond}\ {}_{a}^{c}P \cup^{\diamond} (T,C) \cup^{\diamond} (T \cup^{\diamond} \{ {}_{a}^{cb} \}, C \setminus^{\diamond} \{t\})\{\![a^{c}/a^{m_{\mu}}]\!\}}$$

$$\mathbf{Open}^{\diamond} \quad \frac{{}_{a}^{b}\mathtt{open}\, m_{\lambda^{\diamond}}.P \in C \qquad {}_{m_{\mu}}^{ab} \in T \qquad (a = m_{\ell'_1} \Rightarrow \ell'_1 \neq \mu)}{(T,C) \mapsto^{\diamond} \delta^{\diamond}\ {}_{a}^{b}P \cup^{\diamond} (T,C) \cup^{\diamond} (T,C)\{\![a^{b}/{m_{\mu}}^{a}]\!\}\{\![c^{a}/c^{m_{\mu}}]\!\}}$$

Table 7: Abstract transitions $\mapsto^{\diamond}$

The abstract semantics is a safe approximation of the collecting semantics. Safeness is stated in classical abstract interpretation style showing that the abstract semantics is an upper approximation of the property we are interested in.

Lemma 5.8 *Let $S_2 \in \mathcal{S}$ and $S^{\natural} \in \mathcal{S}^{\natural}$. We have*

$$\alpha^{\diamond}(\Psi_{S_2}(S^{\natural})) \subseteq^{\diamond} \Psi^{\diamond}_{\alpha^{\diamond}(S_2)}(\alpha^{\diamond}(S^{\natural})).$$

The proof of Lemma 5.8 is shown in the Appendix B.1. The proof exploits two main properties which show the safeness of: the abstract normalisation function $\delta^{\diamond}$ (Proposition B.7) with respect to δ; the abstract transitions $\mapsto^{\diamond}$ with respect to the concrete transitions $\mapsto$ (Lemma B.8).

Theorem 5.9 (Safeness) *Let P be a well-labeled process. We have*

$$\alpha^{\diamond}(\mathfrak{S}_{Coll}[\![P]\!]) \subseteq^{\diamond} \mathfrak{S}_{Coll^{\diamond}}[\![P]\!].$$

Proof: By Definitions 5.7 and 4.9 we have to show that

$$\alpha^{\diamond}(lfp\ \Psi_{\delta\ @P}) \subseteq^{\diamond} lfp\ \Psi^{\diamond}_{\alpha^{\diamond}(\delta\ @P)}.$$

This follows from Lemma 5.8 using Theorem 2.2.

$\square$

Examples. We present some examples to summarize the most interesting aspects of the abstraction. The following example explains more in details the role of indexes in the abstraction. Any labeling of a process P respecting the requirements of Definition 4.2 is enough to have a correct normal semantics of P. However, the choice of labels has dramatic consequences on the precision of the abstraction. Hence, a convenient annotation schema consists of keeping all labels distinct also up to re-indexing.

Example 5.10 *Consider the processes*

$$P_1 = n_{\ell_1}[\mathtt{in}\,k_\mu] \mid n_{\ell_2}[\mathtt{in}\,m_\gamma] \mid m_\lambda[0]$$

$$P_2 = n_\beta[\mathtt{in}\,k_\mu] \mid n_\epsilon[\mathtt{in}\,m_\gamma] \mid m_\lambda[0]$$

where $\{\mu, \gamma, \lambda, \beta, \epsilon\}$ are distinct also up to re-indexing and are not of the form ℓ_i. We have

$$\mathfrak{S}_{Coll}[\![P_1]\!] = (\{\ {n_{\ell_1}}^{@},\ {n_{\ell_2}}^{@},\ {m_\lambda}^{@},\ {n_{\ell_2}}^{m_\lambda}\},\{\ {}^{n_{\ell_1}}\mathtt{in}\,k_\mu,\ {}^{n_{\ell_2}}\mathtt{in}\,m_\gamma\})$$

$$\mathfrak{S}_{Coll}[\![P_2]\!] = (\{\ {n_\beta}^{@},\ {n_\epsilon}^{@},\ {m_\lambda}^{@},\ {n_\epsilon}^{m_\lambda}\},\{\ {}^{n_\beta}\mathtt{in}\,k_\mu,\ {}^{n_\epsilon}\mathtt{in}\,m_\gamma\}).$$

Obviously the processes P_1 and P_2 are equivalent up to renaming of labels. Notice that only ambients n_{ℓ_2} and n_ϵ may end up inside m_λ. In the abstract semantics we have (for readability we use $\{\mu, \gamma, \lambda, \beta, \epsilon\}$ for the corresponding labels with multiplicity one)

$$\mathfrak{S}_{Coll^\diamond}[\![P_1]\!] = (\{\ {n_{\ell_\omega}}^{@\top},\ {m_\lambda}^{@\top},\ {n_{\ell_\omega}}^{m_\lambda^{@}}\},\{\ {}^{{n_{\ell_\omega}}^{@}}\mathtt{in}\,k_\mu,\ {}^{{n_{\ell_\omega}}^{@}}\mathtt{in}\,m_\gamma\})$$

$$\mathfrak{S}_{Coll^\diamond}[\![P_2]\!] = (\{\ {n_\beta}^{@\top},\ {n_\epsilon}^{@\top},\ {m_\lambda}^{@\top},\ {n_\epsilon}^{m_\lambda^{@}}\},\{\ {}^{{n_\beta}^{@}}\mathtt{in}\,k_\mu,\ {}^{{n_\epsilon}^{@}}\mathtt{in}\,m_\gamma\}).$$

Due to a different choice of labels the results reported by the analysis are different: for process P_1 the two ambients with name n are both represented by n_{ℓ_ω}; while for process P_2 ambients n_β and n_ϵ are keep distinct. Consequently, the analysis of P_1 is less precise; it says that both n_{ℓ_1} and n_{ℓ_2} may end up inside m.

The following example shows the analysis of the process considered in Example 4.11, where an ambient n moves inside an ambient k, and then is opened unleashing no capability of movement inside k. Due to the combination of the multiplicity and of the partial topology, the analysis is sufficiently precise to capture what is executed inside n *before* and *after* n is opened. In particular, it argues that the capability of movement $\mathtt{in}\,k$ has been consumed when n is opened. Consequently, it says that ambient k acquires, when opens the mobile ambient n, only an immobile process.

Example 5.11 *Consider the process shown in Example 4.11 (see the semantics in Figure 2)*

$$P = n_\lambda[\mathtt{in}\,k_\epsilon.\,m_\zeta[Q_2]] \mid k_\mu[\mathtt{open}\,n_\beta.\,Q_1].$$

We discuss the abstract semantics of the process P assuming that $Q_1 = Q_2 = 0$ and that the indexed labels $\{\lambda, \epsilon, \zeta, \mu, \beta\}$ are distinct also up to re-indexing. The initial abstract state representing the process P is $S_0^\diamond = (T_0^\diamond, C_0^\diamond)$ where

$$T_0^\diamond = \{\ {n_\lambda}^{@^\top},\ {k_\mu}^{@^\top}\}$$

$$C_0^\diamond = \{\ {}^{{n_\lambda}^{@}}\mathtt{in}\, k_\epsilon.\, m_\zeta[0],\ \ {}^{{k_\mu}^{@}}\mathtt{open}\, n_\beta\}.$$

By applying rule **In**$^\diamond$ *we have a transition $S_0^\diamond \mapsto^\diamond S_1^\diamond$ where $S_1^\diamond = (T_1^\diamond, C_0^\diamond)$ and*

$$T_1^\diamond = T_0^\diamond \cup \{\ {m_\zeta}^{{n_\lambda}^{k_\mu}},\ {n_\lambda}^{{k_\mu}^{@}}\}.$$

The capability $\mathtt{in}\, k$ is exercised inside n, when n and k are within @. Its execution modifies the abstract topology: (i) ${n_\lambda}^{{k_\mu}^{@}}$ is added to model the movement of n inside k; (ii) ${m_\zeta}^{{n_\lambda}^{k_\mu}}$ is added because the continuation of $\mathtt{in}\, k$ ($m_\zeta[0]$) becomes executable after n has moved inside k. Notice that the capability $\mathtt{in}\, k$ has multiplicity one, and thus ${}^{{n_\lambda}^{k_\mu}}\mathtt{in}\, k_\epsilon.\, m_\zeta[0]$ does not belong to the abstract configuration. This says that $\mathtt{in}\, k$ has been consumed when n is within k.

We observe that only rule **In**$^\diamond$ *can be applied in state $S_0^\diamond$; the capability $\mathtt{open}\, n$ cannot be exercised since n is not within k(${n_\lambda}^{{k_\mu}^{@}} \notin T_0^\diamond$). Rule* **Open**$^\diamond$ *becomes instead executable in state $S_1^\diamond$ where k is one of the fathers of n.*

By applying rule **Open**$^\diamond$ *we have a transition $S_1^\diamond \mapsto^\diamond S_2^\diamond$ where $S_2^\diamond = (T_2^\diamond, C_0^\diamond)$ and*

$$T_2^\diamond = T_1^\diamond \cup \{\ {m_\zeta}^{{k_\mu}^{@}}\}.$$

The execution of $\mathtt{open}\, n$ inside k produces the unleashing inside k only of those processes and ambients which are contained inside n, when n is within k. Those processes and ambients are determined using the partial topology. Since ${m_\zeta}^{{n_\lambda}^{k_\mu}} \in T_1^\diamond$, then ambient m ends up inside k, that is ${m_\zeta}^{{k_\mu}^{@}}$ is added to the abstract topology. No other ambient or process is acquired by k, in particular the process $\mathtt{in}\, k_\epsilon.\, m_\zeta[0]$, which can be executed inside n only when n is inside @.

Therefore, the abstract semantics is (depicted also in Figure 5)[8] $\mathfrak{S}_{Coll^\diamond}[\![P]\!] = S_2^\diamond$. The analysis shows that: k is an immobile ambient (there are no capabilities of movement inside k); n is a mobile ambient (the capability $\mathtt{in}\, k$ is exercised inside n); ambient n unleashes, when opened, an immobile process (that is $m_\zeta[0]$). As we have explained above both the labels with multiplicity and the partial topology are needed to achieve this very accurate prediction.

The following example shows the analysis of the processes discussed in Example 4.12 and clarifies how the replicated processes are identified by the abstraction.

[8]As usual we have omitted labels to simplify the picture.

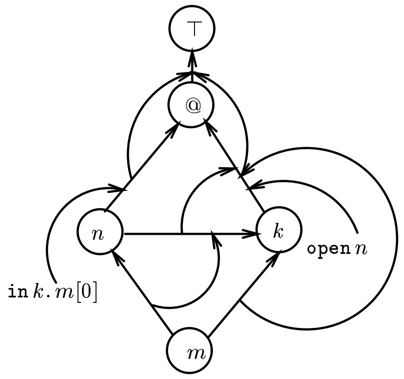

Figure 5: The abstract semantics of P

Example 5.12 *Consider the process $Q = n_\lambda[\mathtt{in}\,n_\gamma]$ of Example 4.12, where $\lambda = \ell_1$ and $\gamma = \ell'_1$. We have for $\lambda^\diamond = \ell_\omega$ and $\gamma^\diamond = \ell'_\omega$,*

$$\mathfrak{S}_{Coll^\diamond}[\![Q]\!] = (\ {n_\lambda}^{@^{\top}},\ \ {}^{{n_\lambda}^{@}}\mathtt{in}\,n_\gamma)$$

$$\mathfrak{S}_{Coll^\diamond}[\![!Q]\!] = (\{\ {n_{\lambda^\diamond}}^{{n_{\lambda^\diamond}}^{n_{\lambda^\diamond}}},\ {n_{\lambda^\diamond}}^{@^{\top}},\ {n_{\lambda^\diamond}}^{{n_{\lambda^\diamond}}^{@}}\},\{\ {}^{{n_{\lambda^\diamond}}^{@}}\mathtt{in}\,n_{\gamma^\diamond},\ \ {}^{{n_{\lambda^\diamond}}^{n_{\lambda^\diamond}}}\mathtt{in}\,n_{\gamma^\diamond},\ {}^{@}!Q\})$$

$$\mathfrak{S}_{Coll^\diamond}[\![!(\nu n_\mu)\,Q]\!] = (\{\ {n_{\lambda^\diamond}}^{{n_{\lambda^\diamond}}^{n_{\lambda^\diamond}}},\ {\hat{n}_{\lambda^\diamond}}^{@^{\top}},\ {\hat{n}_{\lambda^\diamond}}^{{\hat{n}_{\lambda^\diamond}}^{@}}\},\{\ {}^{{\hat{n}_{\lambda^\diamond}}^{@}}\mathtt{in}\,\hat{n}_{\gamma^\diamond},\ \ {}^{{\hat{n}_{\lambda^\diamond}}^{\hat{n}_{\lambda^\diamond}}}\mathtt{in}\,\hat{n}_{\gamma^\diamond},\ {}^{@}!(\nu n_\mu)\,Q\})$$

The labels with multiplicity permit to distinguish process $!Q$ from process Q. In the abstract semantics of Q the label of n is ℓ_1, which forbids the movement of n inside itself (see rule **In**$^\diamond$*). Conversely, in the abstract semantics of $!Q$ the unfolding of recursion produces a label ℓ_ω for n and a label ℓ'_ω for $\mathtt{in}\,n$, which force this movement (see rule* **In**$^\diamond$*. Consequently, we have both ${n_{\lambda^\diamond}}^{{n_{\lambda^\diamond}}^{@}}$ and ${n_{\lambda^\diamond}}^{{n_{\lambda^\diamond}}^{n_{\lambda^\diamond}}}$ in the abstract topology. Recall that the unfolding of replication produces multiple copies of n, which may interact with each other as we have shown in Figure 3. In particular, any copy of n may enter within another copy of n which is top level (inside @.) This shows a subtle difference between these two statements: ${n_{\lambda^\diamond}}^{{n_{\lambda^\diamond}}^{@}}$ is necessary to have a safe approximation of the concrete semantics; instead ${n_{\lambda^\diamond}}^{{n_{\lambda^\diamond}}^{n_{\lambda^\diamond}}}$ is an approximation due to the multiplicity ω of capability $\mathtt{in}\,n$.*

The analysis infers the same information for both processes $!Q$ and $!(\nu n_\mu)\,Q$. In the abstract domain the distinct names $\hat{n}_1, \hat{n}_2 \ldots$, produced by the unfolding of replication, are represented by $\hat{n}$. Thus, the ambients $\hat{n}$ interact with each other (see rule **In**$^\diamond$*).*

6 A Second Abstraction

On top of the previous abstraction, we define a new abstraction, aimed at computing more efficiently an approximation of a weaker property. We want to know the following information about all the states reachable from the initial state representing a process

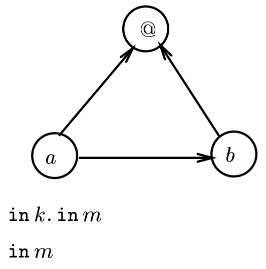

Figure 6: The state S°

P: for each ambient n, which ambients and capabilities may be contained (at top level). inside n. The abstraction is simply obtained from the analysis of Section 5 by dropping the multiplicity from labels and the partial topology from the topology and the configuration.

Consider for instance the states (4) and (5) shown at the beginning of Section 5

$$S_1 = (\{\ {}_{a}^{@},\ {}_{b}^{@}\},\ {}^{a}\mathtt{in}\,k_\mu.\,\mathtt{in}\,m_\gamma)$$

$$S_2 = (\{\ {}_{a}^{b},\ {}_{b}^{@}\},\ {}^{a}\mathtt{in}\,m_\gamma).$$

The set of states $\{S_1, S_2\}$ is represented by the following abstract state which is simply their union (depicted also in Figure 6)[9]

$$S^\circ = (\{\ {}_{a}^{b},\ {}_{b}^{@},\ {}_{a}^{@}\},\{\ {}^{a}\mathtt{in}\,m_\gamma,\ {}^{a}\mathtt{in}\,k_\mu.\,\mathtt{in}\,m_\gamma\}).$$

The abstract configuration says that both $\mathtt{in}\,m_\gamma$ and $\mathtt{in}\,k_\mu.\,\mathtt{in}\,m_\gamma$ are active processes inside a. With respect to the abstraction of Section 5, shown in Figure 4, we lose the information that the former is executable, when a is inside b; while the latter is executable, when a is inside @. Similarly for the topology.

Moreover, consider the states (6) and (7) shown at the beginning of Section 5

$$S_1 = (\ {}_{n_{\ell'_1}}^{@},\ {}^{n_{\ell'_1}}\mathtt{open}\,m_{\ell_1})$$

$$S_2 = (\ {}_{n_{\ell'_2}}^{@},\{\ {}^{n_{\ell'_2}}\mathtt{open}\,m_{\ell_1},\ {}^{n_{\ell'_2}}\mathtt{open}\,m_{\ell_2}\}).$$

In the new abstraction S_1 and S_2 are represented by the same abstract state

$$S^\circ = (\ {}_{n_{\ell'}}^{@},\ {}^{n_{\ell'}}\mathtt{open}\,m_{\ell}).$$

Therefore, we lose the ability to distinguish one occurrence from multiple occurrences of an object.

[9]As usual we have omitted labels to simplify the picture.

Abstract domain. The abstract labels are $\mathcal{L}$ and the abstract names are $\mathcal{N} \cup \widehat{\mathcal{N}}$. The relation between names and labels is given precisely by function $H_{\mathcal{L}} : \mathcal{L} \to \widehat{\mathcal{N}}$. We use $\mathcal{A}^\circ$ (ranged over by $a^\circ, b^\circ, c^\circ \ldots$) for the set of abstract labeled names n_ℓ, such that $n \in \mathcal{N} \cup \widehat{\mathcal{N}}$ and $\ell \in \mathcal{L}$, augmented with the symbol @. The abstract processes are built according to the syntax of Definition 4.1 over names $\mathcal{N} \cup \widehat{\mathcal{N}}$ and labels $\mathcal{L}$. As usual we use $\mathcal{P}^\circ$ and $\mathcal{AP}^\circ$ to denote the set of abstract processes and active abstract processes.

Definition 6.1 (Abstract States) *An* abstract state S° *is a pair* (T°, C°) *where*

1. $T^\circ \in \wp((\mathcal{A}^\circ \setminus \{@\}) \times \mathcal{A}^\circ)$;
2. $C^\circ \in \wp(\mathcal{A}^\circ \times \mathcal{AP}^\circ)$.

In an abstract state (T°, C°) we call T° the topology and C° the configuration. We assume that all the previously defined notions on states and processes are adapted to abstract states and processes in the expected way. We use $\mathcal{S}^\circ$ to denote the set of abstract states.

The abstract domain is given by the abstract states ordered by inclusion [10].

Definition 6.2 *The* abstract domain *is* $\langle \mathcal{S}^\circ, \subseteq \rangle$.

In the following we may omit the over-script $-^\circ$ for any syntactic category, when the meaning is clear from the context.

The Galois connection. The relation between the abstract domain of Definition 5.4 and the abstract domain of Definition 6.2 is established by a Galois connection (see Definition 2.1). An abstract state is abstracted, as explained at the beginning of the section, by dropping both the multiplicity from labels and the partial topology. To this purpose, we use a renaming $\rho^\circ : \mathcal{L}^\diamond \to \mathcal{L}$, such that $\rho^\circ(\ell_1) = \rho^\circ(\ell_\omega) = \ell$.

Definition 6.3 *Let* $(T^\diamond, C^\diamond) \in \mathcal{S}^\diamond$ *and* $S^\circ \in \mathcal{S}^\circ$. *We define* $\alpha^\circ : \mathcal{S}^\diamond \to \mathcal{S}^\circ$ *and* $\gamma^\circ : \mathcal{S}^\circ \to \mathcal{S}^\diamond$ *as follows*

1. $\alpha^\circ((T^\diamond, C^\diamond)) = (\{ \, {}_a^{\;b} \mid {}_a^{\;b^c} \in T^\diamond\}, \{ \, {}^aP \mid {}_a^{\;b}P \in C^\diamond\})\rho^\circ$;
2. $\gamma^\circ(S^\circ) = \bigcup^\diamond_{S^\diamond \in \{S^{\diamond\prime} \mid \alpha^\circ(S^{\diamond\prime}) \subseteq S^\circ\}} S^\diamond$.

The pair of functions defined above is a Galois connection.

Theorem 6.4 *The pair of functions* $(\alpha^\circ, \gamma^\circ)$ *is a Galois connection between* $\langle \mathcal{S}^\diamond, \subseteq^\diamond \rangle$ *and* $\langle \mathcal{S}^\circ, \subseteq \rangle$.

[10] As usual we assume $\subseteq$ and $\cup$ defined component-wise.

The proof of Theorem 6.4 is shown in the Appendix B.2.

Abstract semantics. The abstract normalisation function $\delta^\circ : \mathcal{A}^\circ \times \mathcal{P}^\circ \to \mathcal{S}^\circ$ is given by the rules of Table 4 with a minor modification. It is enough to replace the concrete labels $\mathcal{L}_I$ with the abstract labels $\mathcal{L}$, that using the substitution the function $H_{\mathcal{L}}$ in place of $H_{\mathcal{L}_I}$.

The abstract transitions are defined by the rules of Table 8. Rule **Bang**$^\circ$ is used to unfold replication; it creates a copy of the replicated process without modifying the labels. The rules **In**$^\circ$, **Out**$^\circ$, **Open**$^\circ$ realise the movements and the opening. They are similar to the corresponding rules of the abstract semantics in Table 7 in the case of multiplicity ω. The only relevant difference is that, due to the removal the partial topology, the conditions to be checked for the execution of capabilities are weaker. For instance, rule **In**$^\circ$ can be applied, whenever ambient a and an ambient with name m have a common father b in the topology. There is no check on the father of b to guarantee that ambients a and m are contained in b at the same time.

Bang$^\circ$	$\dfrac{{}^{a}!P \in C}{(T,C) \mapsto^\circ \delta^\circ\ {}^{a}P\ \cup\ (T,C)}$
In$^\circ$	$\dfrac{{}^{a}\mathtt{in}\, m_\ell.\, P \in C \qquad {}_{a}{}^{b},\ {}_{m_{\ell'}}{}^{b} \in T}{(T,C) \mapsto^\circ \delta^\circ\ {}^{a}P\ \cup\ (T \cup \{\ {}_{a}{}^{m_{\ell'}}\}, C)}$
Out$^\circ$	$\dfrac{{}^{a}\mathtt{out}\, m_\ell.\, P \in C \qquad {}_{a}{}^{m_{\ell'}},\ {}_{m_{\ell'}}{}^{b} \in T}{(T,C) \mapsto^\circ \delta^\circ\ {}^{a}P\ \cup\ (T\ \cup\ \{\ {}_{a}{}^{b}\}, C)}$
Open$^\circ$	$\dfrac{{}^{a}\mathtt{open}\, m_\ell.\, P \in C \qquad {}_{m_{\ell'}}{}^{a} \in T}{(T,C) \mapsto^\circ \delta^\circ\ {}^{a}P\ \cup\ (T,C)\ \cup\ (T,C)\{[a/m_{\ell'}]\}}$

Table 8: Abstract transitions $\mapsto^\circ$

The abstract semantics is defined as follows.

Definition 6.5 (The abstract semantics) *Let $S_1^\circ, S_2^\circ \in \mathcal{S}^\circ$, and let P be a well-labeled process. We define $\mathfrak{S}_{Coll^\circ}[\![P]\!] =$ lfp $F^\circ(\alpha^\circ(\alpha^\diamond(\delta\ {}^{@}P)))$ for the function $F^\circ : \mathcal{S}^\circ \to (\mathcal{S}^\circ \to \mathcal{S}^\circ)$ such that $F^\circ(S_2^\circ) = \Psi^\circ_{S_2^\circ}$ and*

$$\Psi^\circ_{S_2^\circ}(S_1^\circ) = S_2^\circ \cup \bigcup_{S^\circ \in \{S_3^\circ \mid S_1^\circ \mapsto^\circ S_3^\circ\}} S^\circ.$$

The abstract semantics defined above is a safe approximation of the abstract semantics of Definition 5.7.

Lemma 6.6 *Let $S_1^\diamond, S_2^\diamond \in \mathcal{S}^\diamond$. We have*

$$\alpha^\circ(\Psi^\diamond_{S_2^\diamond}(S_1^\diamond)) \subseteq \Psi^\circ_{\alpha^\circ(S_2^\diamond)}(\alpha^\circ(S_1^\diamond)).$$

The proof of Lemma 6.6 is shown in the Appendix B.2. As before, the proof relies on the safeness of the abstract normalisation function δ° with respect to $\delta^\diamond$ (Proposition B.13), and the abstract transitions $\mapsto^\circ$ with respect to the transitions $\mapsto^\diamond$ (Lemma B.14).

Theorem 6.7 (Safeness) *Let P be a well-labeled process. We have*

$$\alpha^\circ(\mathfrak{S}_{Coll^\diamond}[\![P]\!]) \subseteq \mathfrak{S}_{Coll^\circ}[\![P]\!].$$

Proof: By Lemma 6.6 and Theorem 2.2 similarly as in Theorem 5.9.

□

It is a well-known result of abstract interpretation that Galois connections are closed under composition. Therefore, an immediate consequence of Theorem 6.7 is that the new abstract semantics is a safe approximation of the collecting concrete semantics.

Corollary 6.8 *Let P be a well-labeled process. We have*

$$\alpha^\circ(\alpha^\diamond(\mathfrak{S}_{Coll}[\![P]\!])) \subseteq \mathfrak{S}_{Coll^\circ}[\![P]\!].$$

Examples. We discuss the differences between the abstraction presented in this section and the abstraction of Section 5. One relevant difference is that the second abstraction does not distinguish between one or many occurrences of an object. Consequently, the second abstraction infers the same information for the processes Q, $!Q$ and $!(\nu n_\mu)\ Q$ discussed in the Examples 5.12 and 4.12. Another loss of information is due to the removal of the partial topology. The following examples explain that, consequently, the ability to argue on the ordering of execution of capabilities is lost.

Example 6.9 *Consider the process of Example 4.11 (see the semantics in Figure 2)*

$$P = n_\lambda[\mathtt{in}\, k_\epsilon.\, m_\zeta[Q_2]] \mid k_\mu[\mathtt{open}\, n_\beta.\, Q_1].$$

Assuming that $Q_1 = Q_2 = 0$, we derive the abstract semantics (depicted also in Figure 7) [11] *$\mathfrak{S}_{Coll^\circ}[\![P]\!] = (T^\circ, S^\circ)$ where (for readability we use $\{\lambda, \gamma, \mu, \epsilon, \zeta, \beta\}$ to denote the corresponding abstract labels without indexes)*

$$T^\circ = \{\ {}_{n_\lambda}{}^{@},\ {}_{m_\zeta}{}^{n_\lambda},\ {}_{k_\mu}{}^{@},\ {}_{n_\lambda}{}^{k_\mu},\ {}_{m_\zeta}{}^{k_\mu},\ {}_{k_\mu}{}^{k_\mu}\}$$

$$C^\circ = \{\ {}^{n_\lambda}\mathtt{in}\, k_\epsilon.\, m_\zeta[0],\ {}^{k_\mu}\mathtt{open}\, n_\beta,\ {}^{k_\mu}\mathtt{in}\, k_\epsilon.\, m_\zeta[0]\}.$$

[11] As usual we have omitted labels to simplify the picture.

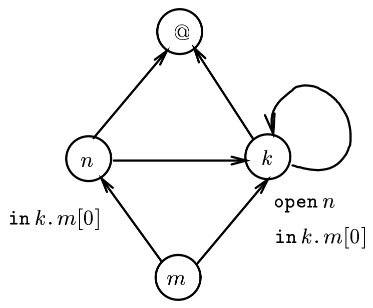

Figure 7: The abstract semantics of process P

The result of the first analysis has been discussed in Example 5.11 (Figure 5). The second analysis is substantially less precise; it is not able to capture that capability $\mathtt{in}\,k$ *has been consumed before opening. Consequently, it says that ambient* k *acquires also* $\mathtt{in}\,k$*, when opens* n*. Also, since the analysis cannot reason on how many occurrences of ambient* k *are present, it says that ambient* k*, by exercising* $\mathtt{in}\,k$*, enters inside itself (see rule* $\mathbf{In}^{\circ}$*). Thus,* k *is reported as mobile ambient.*

Example 6.10 *Consider the process* $P = P_1 \mid P_2 \mid P_3$*, where* $P_1 = !n_\lambda[\mathtt{in}\,m_\mu.\,\mathtt{in}\,k_\zeta]$*,* $P_2 = !m_\beta[0]$ *and* $P_3 = !k_\gamma[0]$*. Assume that labels* $\{\lambda, \mu, \zeta, \beta, \gamma\}$ *are distinct also up to re-indexing. In the first abstraction we have* $\mathfrak{S}_{Coll^\diamond}[\![P]\!] = (T^\diamond, C^\diamond)$ *where (for readability we use* $\{\lambda, \mu, \zeta, \beta, \gamma\}$ *to denote the corresponding abstract labels annotated with* ω*)*

$$T^\diamond = \{\ {n_\lambda}^{@^\top},\ {k_\gamma}^{@^\top},\ {m_\beta}^{@^\top},\ {n_\lambda}^{{m_\beta}^{@}}\}$$

$$C^\diamond = \{\ {}^{{n_\lambda}^{@}}\mathtt{in}\,m_\mu.\,\mathtt{in}\,k_\zeta,\ \ {}^{{n_\lambda}^{m_\beta}}\mathtt{in}\,m_\mu.\,\mathtt{in}\,k_\zeta,\ \ {}^{{n_\lambda}^{m_\beta}}\mathtt{in}\,k_\zeta,\ {}^{@}P_1,\ {}^{@}P_2,\ {}^{@}P_3\}.$$

The analysis shows that capability $\mathtt{in}\,k$ *is not exercised inside* n*. In fact, the partial topology says that, it is executable only when* n *has moved inside* m*. Ambient* k *does not move and, consequently, cannot be within* m*.*

In the second abstraction we have (for readability we use $\{\lambda, \mu, \zeta, \beta, \gamma\}$ *for the corresponding abstract labels without indexes),*

$$\mathfrak{S}_{Coll^\circ}[\![P]\!] = (\{\ {n_\lambda}^{@},\ {k_\gamma}^{@},\ {m_\beta}^{@},\ {n_\lambda}^{m_\beta},\ {n_\lambda}^{k_\gamma}\}, \{\ {}^{n_\lambda}\mathtt{in}\,m_\mu.\,\mathtt{in}\,k_\zeta,\ {}^{n_\lambda}\mathtt{in}\,k_\zeta,\ {}^{@}P_1,\ {}^{@}P_2,\ {}^{@}P_3\}).$$

The analysis predicts that $\mathtt{in}\,k$ *can be executed, because* n *and* k *have @ as a common father. Due to the removal of the partial topology, does not detect that* $\mathtt{in}\,k$ *becomes executable inside* n only after *the movement inside* m*.*

It is worth noticing that the result of the first analysis is not optimal, meaning that

$$\alpha^\diamond(\mathfrak{S}_{Coll}[\![P]\!]) \subset^\diamond \mathfrak{S}_{Coll^\diamond}[\![P]\!].$$

For instance, in the abstract semantics we have ${{}^{n_\lambda}}^{m_\beta}\,\mathtt{in}\,m_\mu.\,\mathtt{in}\,k_\zeta$ *which says that* $\mathtt{in}\,m$ *is still executable inside* n*, when* n *is within* m*. Instead, in any instance of ambient* n *capability* $\mathtt{in}\,m$ *has been obviously consumed at that time. This approximation is due to the removal of the indexes, which in this case identifies all ambients* n *and all capabilities* $\mathtt{in}\,m$ *(see rule* $\mathbf{Bang}^{\diamond}$*).*

The abstraction presented in this section uses an abstract domain analogous to that of the CFA proposed in [24]. Our analysis is however more precise as the following example shows.

Example 6.11 *Consider the process* $P = n_\lambda[\mathtt{in}\,m_\mu.\,\mathtt{in}\,k_\zeta] \mid k_\gamma[m_\beta[0]]$*. We obtain (for readability we use* $\{\lambda,\mu,\zeta,\beta,\gamma\}$ *to denote the corresponding abstract labels without indexes)*

$$\mathfrak{S}_{Coll^\circ}[\![P]\!] = (\{\ {n_\lambda}^{@},\ {k_\gamma}^{@},\ {m_\beta}^{k_\gamma}\},\{\ {}^{n_\lambda}\mathtt{in}\,m_\mu.\,\mathtt{in}\,k_\zeta\}).$$

The analysis shows that the system is deadlocked: neither capability $\mathtt{in}\,m$ *nor capability* $\mathtt{in}\,k$ *can be executed. The former because ambient* m *is not a sibling of* n*, the latter because it is guarded by* $\mathtt{in}\,m$*.*

The analysis of [24] considers the effect of the continuation of a capability regardless of whether the capability may be exercised. Consequently, for process P *it predicts that* n *moves inside* k *and, consequently, also inside* m*.*

7 Applications to Security

We show some examples to demonstrate that the analyses we have proposed can be used to establish interesting security properties. In particular, we show the results obtained using the abstraction of Section 5 for two simple examples found in the literature [16, 5]. Another typical example is the firewall protocol, which can be proved correct also by applying the weaker analysis of Section 6. This example in fact can be checked also by the CFA of [24].

Example 7.1 (Secrecy)

Degano et al. [16] consider a property of secrecy based on a standard classification of ambients into *untrusted* and *trusted*. Secrecy of data is preserved if an untrusted ambient can never open a trusted ambient, since opening an ambient gives indeed access to its content. They show that the property holds for the following system (actually for its SA version)

$$SYS = (\nu\ mail)\ (a[mail[\mathtt{out}\,a.\,\mathtt{in}\,b.\,msg[\mathtt{out}\,mail.\,D]]]) \mid b[\mathtt{open}\,msg] \mid C.$$

The pilot ambient *mail* goes out of a, and then enters b. Once there, *msg* goes out of *mail*, and b acquires the data D by opening *msg*. When the data D is secret, it is essential to guarantee that no ambient can open msg except for the designated receiver b. Assume that $\{b, msg\}$ is the set of trusted ambients, and that all the others (including @) are untrusted. We wish to prove that no untrusted ambient can open msg.

Assume that the parallel process C is $\mathtt{open}\, msg$ meaning that the untrusted ambient @ tries to read the data D. By applying the analysis of Section 5 we derive $\mathfrak{S}_{Coll^\diamond}[\![SYS]\!] = (T^\diamond, S^\diamond)$ where[12]

$$T^\diamond = \{\ {a}^{@^\top},\ {mail}^{a^@},\ {b}^{@^\top},\ {mail}^{@^\top},\ {mail}^{b^@},\ {msg}^{mail^b},\ {msg}^{b^@}\}$$

$$C^\diamond = \begin{array}{l}\{\ {}_{@}^{\top}\mathtt{open}\, msg,\ \ {}_{b}^{@}\mathtt{open}\, msg,\ \ {}_{mail}^{a}\mathtt{out}\, a.\, \mathtt{in}\, b.\, msg[\mathtt{out}\, mail.\, D],\\ \ {}_{mail}^{@}\mathtt{in}\, b.\, msg[\mathtt{out}\, mail.\, D],\ \ {}_{msg}^{mail}\mathtt{out}\, mail.\, D,\ \ {}_{msg}^{b}D,\ \ {}_{b}^{@}D\}\end{array}$$

This result shows that only b can open the messenger ambient msg. Consequently the secret data may end up in b only, as shown by ${}_{msg}^{b}D$ and ${}_{b}^{@}D$. Both the partial topology and the multiplicity are needed to achieve this result. The main observations concerning the analysis are:

- the capability $\mathtt{open}\, msg$ cannot be exercised in @, because msg cannot end up within @. This is reported by the abstract topology, in particular by ${msg}^{mail^b}$ and ${msg}^{b^@}$;
- the execution of the capability $\mathtt{out}\, mail$ inside msg, lets msg go only inside ambient b, as msg can be contained in $mail$, only when $mail$ is within b (see rule **Out**$^\diamond$). The latter condition is modeled by ${msg}^{mail^b}$;
- the multiplicity of capabilities $\mathtt{out}\, a$ and $\mathtt{in}\, a$ is used to conclude that msg can be contained in $mail$ only when $mail$ is within b (see rules **Out**$^\diamond$ and **In**$^\diamond$).

The analyses of [24] and of Section 6 are too weak to prove the secrecy of this system. They predict that msg, when goes out of $mail$, may end up in any of the fathers of $mail$, namely a, b and @. This example shows that the analysis of Section 5 gives results comparable to those obtained for SA in [16]. In SA, however, it is easier to get such an accurate prediction, because coactions control precisely when and where capabilities can be exercised.

Example 7.2 (Security Boundaries)

Braghin et al. [5] study multilevel security for Mobile Ambients. The original idea is that of introducing *boundary* ambients to protect *high level* information; *high level* data can be contained either in *boundary* ambients or in *low level* ambients which do not escape boundaries. They refine the analysis of [24] to establish more precisely the property above. In particular, they show the following motivating system

$$SYS = a[send[\mathtt{out}\, a.\, \mathtt{in}\, b \mid hdata[\mathtt{in}\, filter]]] \mid b[\mathtt{open}\, send] \mid filter[\mathtt{in}\, send] \mid \mathtt{open}\, filter.$$

[12]We have omitted the abstract labels for readability, they have all multiplicity 1.

The boundary ambient *send* carries the high level ambient *hdata* out of a. Then, it lets a possibly low level *filter* ambient enter, and then it enters the boundary ambient b. Once there, it is dissolved. The system satisfies the security property stated above: *hdata* is always within a boundary ambient (either *send* or b) or within the low level ambient *filter*. Notice that the ambient *filter* does not carry the ambient *hdata* out of the boundary b.

By applying the analysis of Section 5 we obtain $\mathfrak{S}_{Coll^\diamond}[\![SYS]\!] = (T^\diamond, S^\diamond)$ where [13]

$$T^\diamond = \begin{array}{l} \{\, a^{@^\top},\ send^{a^@},\ filter^{@^\top},\ hdata^{send^a},\ b^{@^\top},\ send^{@^\top},\ hdata^{send^@},\ filter^{send^@},\ filter^{send^b}, \\ \quad send^{b^@},\ hdata^{send^b},\ hdata^{filter^{send}},\ hdata^{b^@},\ hdata^{filter^b},\ filter^{b^@} \} \end{array}$$

$$C^\diamond = \begin{array}{l} \{\ {}^{@^\top}\mathtt{open}\, filter,\ \ {}^{hdata^{send}}\mathtt{in}\, filter,\ \ {}^{filter^@}\mathtt{in}\, send, \\ \quad {}^{send^a}\mathtt{out}\, a.\, \mathtt{in}\, b,\ \ {}^{send^@}\mathtt{in}\, b,\ \ {}^{@^\top}\mathtt{in}\, send,\ \ {}^{b^@}\mathtt{open}\, send,\ \ {}^{hdata^b}\mathtt{in}\, filter \} \end{array}$$

The analysis shows that the security property holds, as the abstract topology shows that *hdata* can be within *filter*, only when *filter* is contained in a boundary ambient, either b or *send*. This is modeled by $hdata^{filter^{send}}$ and $hdata^{filter^b}$.

The analysis of [24] as well as the analysis of Section 6 identify, instead, a potential (but practically impossible) attack. Since they do not use the partial topology, they cannot capture that *hdata* enters inside *filter*, only when *filter* is within either *send* or b. Consequently, they predict that *hdata* may end up inside the low level ambient @ as a consequence of the execution of $\mathtt{open}\, filter$.

8 Conclusions and Related Works

We have proposed an abstract interpretation framework for MA based on the normal semantics. The normal semantics uses an explicit representation of the hierarchical structure of processes, in terms of topology and configuration. This representation is more viable for abstraction than the standard reduction semantics. The normal semantics can be compared with the *Gamma* semantic framework for concurrency of [4]: it shares its view of symmetry and locality of interaction, and is based on an explicit representation of multisets.

In the abstract interpretation framework we have derived two safe approximations of the run-time topological structure of processes. To show that these analyses are effective program analysers, it is worth discussing their computational complexity. By restricting the attention to a process P of size n, in the first case the topology of the greatest state contains at most $O(n^3)$ elements and the configuration at most $O(n^3)$ elements. Hence, the iterations before reaching the fixed-point are at most $O(n^3)$. Any iteration has complexity $O(n^5)$, because it requires to check at most $O(n^2)$ conditions for any element of the configuration. Similarly, in the second case we have at most $O(n^2)$ iterations,

[13]We have omitted the abstract labels for readability, they have indeed all multiplicity 1.

where any iteration has complexity $O(n^3)$. Therefore, it is not difficult to devise a naive implementation of the first analysis in $O(n^8)$ and of the second one in $O(n^5)$ by using standard algorithms.

In the last few years there has been a growing interest in the analysis of MA (and its variants) and several CFA in Flow Logic style [24, 25, 26, 16, 5] have been proposed. The analysis of Section 6 is a refinement of the 0-CFA of [24]. The CFA of [24] is less precise, as shown by Example 6.11, and can be obtained in our framework by weakening the conditions on the execution of the continuation of a capability in the rules of Table 8. We refer the reader to [23] for the formal comparison of the two approaches.

The analysis of Section 5 combines together the information about the number of occurrences of objects and the contextual information (i.e. the partial topology). The idea of using the partial topology has been inspired by the 1-CFA of [16] for Safe Ambients. The integration of these two aspects gives accurate predictions as shown by the Examples 5.11, 7.1 and 7.2. These systems are interesting because the considered properties require to have a detailed information about the local process of an ambient, when this is ready to engage into an interaction of opening or movement. We are not aware of similar results in setting of MA apart from those obtained by more complex exponential technique, which use sophisticated information about the context or a sort of causality information [26, 1]. For SA instead the static techniques are more precise due to the presence of coactions. The 1-CFA of [16] for SA, for instance, is simpler than our analysis and is sufficiently precise to prove the secrecy property for the SA process corresponding to that of Example 7.1.

It is worth mentioning that we have introduced the occurrence counting information in the analysis of Section 5 to fruitfully exploit the partial topology. This information is crucial to predict when capabilities may be consumed. The use of the partial topology without that of multiplicity would give limited benefits (see for instance Example 6.10). Other approaches have been proposed to more profitably exploit the information about the number of objects. For instance, Hansen et al [25] show that the 0-CFA of [24] can be derived, by abstract interpretation, from a new more precise and exponential CFA. The refined CFA uses sets of abstract states rather than abstract states and a relational occurrence counting analysis, meaning that the number of occurrences is not counted globally (as in the abstraction of Section 5), but inside any ambient. The use of abstract interpretation in [25] shows several advantages: the CFA's are compared in terms of precision by construction and the properties (in particular the safeness) of the former one are directly derived from those of the latter one. This is precisely what we obtain with the abstraction of Section 6.

Although the interplay between abstract interpretation and CFA in Flow Logic style is not fully understood, these techniques are undoubtedly very similar from an algorithmic point of view and also their specifications are strictly related. For instance, the constraints, which specify the CFA of [24], could be derived by abstract interpretation in our framework; conversely it seems that constraints in Flow Logic style could be given corresponding to the analysis of Section 6. Having said that, it is clear that the approach of [25] is very close to ours. We believe, however, that this paper proposes another original and interesting contribution with respect to the proposal of [25]: the definition of a general abstract interpretation framework, based on the normal semantics. The normal semantics simplifies the development of analyses by means of abstract interpretation; for instance, the derivation of the analysis of Section 6 is rather straightforward once the abstract domain, namely the property we want to compute, has been chosen. Moreover, the derivation of analyses from the normal semantics can be done using standard abstract interpretation techniques to refine and combine domains.

By the time the full version of this paper has been completed, another paper [18] has appeared, which proposes an abstract interpretation framework based on a non-standard semantics similar to the normal semantics. The shape of states and of labels is however slightly different and permits to define an interesting non-uniform analysis where recursive instances of agents are kept distinct. Another CFA that refines the analysis of [24] has been recently proposed in [5]. This work is motivated by the system of Example 7.2 for which the property of multi-level security cannot be established using the approach of [24]. We have shown that this example can be handled also by our analysis. A formal comparison is difficult as the CFA of [5] is designed to establish specifically the property of multi-level security.

This work is part of a project aimed at studying the relationship among abstract interpretation, CFA and types. We believe that the formalisation also of types (for instance of [8, 7]) in an abstract interpretation setting would be very interesting. First, this way we could formally compare the expressive power of CFA's and types, integrate them, understand the pros and cons of each approach, and possibly for which class of properties one method is more adequate than another. Moreover, the development of types as abstract interpretations of a denotational semantics has given very promising results for functional languages [15]. This approach gives in particular more accurate type inference algorithms, based on abstract fixed-point computations and widening operators, and more expressive type systems. It would be interesting to apply this approach also to MA starting for instance from the recent "logical" denotational semantics for higher-order MA of [12]. We leave this investigation to future work. Notice that the comparison with types requires to extend the analyses to the full language with communication. A first step toward this extension has been done by Feret [18], which considers communication of names only. This extension deserves undoubtedly further investigations especially for the analysis of Section 5.

References

[1] Torben Amtoft. Causal Type System for Ambient Movements. Submitted for publication, 2002.

[2] Torben Amtoft and Assaf J. Kfoury and Santiago M. Pericas-Geertsen. What are Polymorphically-Typed Ambients? Proceedings of ESOP'01, LNCS 2028, pages 206–220. Springer Verlag, 2001.

[3] Torben Amtoft and Assaf J. Kfoury and Santiago M. Pericas-Geertsen. Orderly Communication in the Ambient Calculus. To appear in *Computer Languages.*

[4] J.-P.Banâtre and D.Le Métayer. The Gamma model and its discipline of programming. *Science of Computer Programming*, 15:55-77, 1990.

[5] C. Braghin, A. Cortesi and R. Focardi *Control Flow Analysis of Mobile Ambients with Security Boundaries.* Proceedings of FMOODS'02. To appear.

[6] M. Bugliesi and G. Castagna *Secure Safe Ambients.* Proceedings of POPL '01, pages 222-235. ACM Press 2001.

[7] L. Cardelli, G. Ghelli and A. Gordon *Mobility types for mobile ambients.* Proceedings of ICALP' 99, LNCS 1644, pages 230-239. Springer Verlag, 1999.

[8] L. Cardelli and A. Gordon *Types for mobile ambients.* Proceedings of POPL' 99, pages 79-92. ACM Press, 1999.

[9] L. Cardelli and G. Ghelli and A. Gordon. *Ambient Groups and Mobility Types.* Proceedings of IFIP TCS 2000, LNCS 1872, pages 333-347. Springer Verlag, 2000.

[10] L. Cardelli and A. Gordon. *Mobile Ambients.* Proceedings of FoSSaCS' 98, LNCS 1378, pages 140-155. Springer-Verlag, 1998.

[11] L. Cardelli, G. Ghelli, and A.D. Gordon. Types for Ambient Calculus. To apper in *Information and Computation.*

[12] M. Coppo and M. Dezani-Ciancaglini. A fully-abstract model for higher-order mobile ambients. Proceedings of VMCAI'02. To appear in LNCS

[13] P. Cousot and R. Cousot, *Systematic Design of Program Analysis Frameworks.* Proceedings of POPL '79, pages 269–282. ACM Press, 1979.

[14] P. Cousot and R. Cousot, *Abstract Interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints*, Proceedings of POPL '77, pages 238–252, ACM Press, 1977.

[15] P. Cousot, *Types as abstract interpretations.* Proceedings of POPL '97, pages 316–331. ACM Press, 1997.

[16] P. Degano, F. Levi and C. Bodei, *Safe Ambients: Control Flow Analysis and Security.* Proceedings of ASIAN'00, LNCS 1961, pages 199-214. Springer Verlag, 2000.

[17] M. Dezani-Ciancaglini and I. Salvo, *Security Types for Mobile Safe Ambients.* Proceedings of ASIAN'00, LNCS 1961, pages 215-236. Springer Verlag, 2000.

[18] J. Feret, *Abstract Interpretation-Based Analysis of Mobile Ambients.* Proceedings of SAS'01. LNCS 2126, pages 412-430. Springer Verlag, 2001.

[19] X. Guan, Y. Yang, and J. You. Typing Evolving Ambients. *Information Processing Letters*, 80(5), 265-270, 2001.

[20] F. Levi and D. Sangiorgi *Controlling Interference in Ambients.* Proceedings of POPL' 00, pages 352-364. ACM Press, 2000.

[21] F. Levi and D. Sangiorgi *Mobile Safe Ambients.* To appear in Toplas, ACM Press.

[22] F. Levi and S. Maffeis *An Abstract Interpretation Framework for Mobile Ambients.* Proceedings of SAS'01, LNCS 2126, pages 395-411. Springer Verlag, 2001.

[23] S. Maffeis, *Analisi Statiche per la Mobilità.* Master thesis, Dipartimento di Informatica, Università di Pisa, July 2000.

[24] F. Nielson, H.R. Nielson, R.R. Hansen and J.G. Jensen *Validating firewalls in mobile ambients.* Proceedings of CONCUR' 99, LNCS 1664, pages 463-477. Springer-Verlag, 1999.

[25] R. R. Hansen, J. G. Jensen, F. Nielson and H. R.Nielson *Abstract Interpretation of Mobile Ambients.* Proceedings of SAS'99, LNCS 1694, pages 135-148, Springer Verlag, 1999.

[26] H. R. Nielson and F. Nielson *Shape Analysis for Mobile Ambients.* Proceedings of POPL' 00, pages 135-148. ACM Press, 2000.

[27] F. Nielson, H. R. Nielson and C. Hankin, *Principles of Program Analysis.* Springer Verlag, 1999.

APPENDIX

A Proofs of Section 4

In this section we show the proof of Theorem 4.5, which formalises the relation between the normal semantics of Section 4 and the standard reduction semantics of Section 3. For convenience we recall its assertion:
Let P be a well-labeled process and $a \in \mathcal{A}$ which is fresh for P.

1. *If $\delta\ {}^aP \mapsto S$, then there exist a well-labeled process Q, such that $\mathcal{E}(P) \rightarrow_{\equiv} \mathcal{E}(Q)$ and $\delta\ {}^aQ = S$;*
2. *If $\mathcal{E}(P) \rightarrow Q$, then there exist a state S and a well-labeled process Q', such that $\delta\ {}^aP \mapsto^* S$, $\delta\ {}^aQ' = S$ and $Q \equiv \mathcal{E}(Q')$.*

Part 1. shows *soundness* and the converse part 2. shows *completeness*. To simplify the proof, which is rather complex, we extend the reduction semantics to well-labeled processes. The reduction semantics for well-labeled processes is designed precisely to be closer to the normal semantics than the standard reduction semantics. Then, we prove both soundness and completeness in two steps: (i) we show the relation between the reductions of well-labeled processes and those of standard unlabeled processes (Lemmas A.2 and A.9); (ii) we show the relation between the reductions of well-labeled processes and the transitions between the states representing them (Lemmas A.17 and A.20).

In the following, to ease the use of induction in the proofs, we assume that also standard unlabeled processes of Definition 3.1 can be defined over names $\mathcal{N} \cup \widehat{\mathcal{N}}_I$.

A.1 Reduction semantics of well-labeled processes

The reduction semantics for well-labeled processes is defined by the rules of Table 9 and is the obvious adaptation of the standard reduction semantics for the unlabeled processes (Tables 1 and 2). The only difference is that in rule (Cong) we adopt a relation $\gg$, which differs substantially from structural congruence $\equiv$ for unlabeled processes (Table 3). In particular,

1. we rule out the analogues of rules (Pref) and (Bang);
2. we assume that rule (Bang-Bang), which realises the fold/unfold of replication, can be applied only in one way, that is to produce a copy of the replicated process and not to remove it.

These choices are motivated by the aim of having a relation $\gg$ which better reflects the normal semantics. More in details, we want that two well-labeled processes, such that $P \gg Q$, are represented by "equivalent" states when translated via δ (see Lemma A.18). The rules (Pref) and (Bang) give problems as, in the normal semantics, some syntactical differences are removed only at execution time. Consider, for instance, two processes $M.P$ and $M.Q$, where $P \equiv Q$. These processes are represented by two different states (assuming a proper labeling) $\delta\ {}^{@}M.P = (\emptyset,\ {}^{@}M.P)$ and $\delta\ {}^{@}M.Q = (\emptyset,\ {}^{@}M.Q)$. The continuations P and Q are translated via function δ only after the execution of capability M.

Rule (Bang-Bang) gives a similar problem, as the unfolding of replication is modeled by a transition (**Bang**) in the normal semantics. Consider for instance two processes $!P$ and $!P \mid P$. These processes are represented by two different states (assuming a proper labeling) $\delta\ {}^{@}!P = S_1 = (\emptyset,\ {}^{@}!P)$ and $\delta\ {}^{@}(!P \mid P) = S_2 = (\emptyset,\ {}^{@}!P) \cup \delta\ {}^{@}P$. We have $S_1 \mapsto S_2$ by rule **Bang**, but obviously $S_2 \not\mapsto S_1$.

The relation $\gg$ for well-labeled processes is defined in Table 10. As we have explained above $\gg$ is not symmetric, as there is only one way of (Bang-Bang) [14]. In rule (Bang-Bang) the labels of the replicated process are re-indexed to guarantee that $new(P) \mid !P$ is a well-labeled process provided that P is well-labeled. To this aim, we use $new(P)$ which is adapted in the obvious way from the definition of *new* over states (see Section 4). Hence, we let $new(P) = P\rho_I$ where: ρ_I is a re-indexing of labels such that $dom(\rho_I) = \Lambda(P)$; $P\rho_I$ is well-labeled; there is no $\lambda \in \Lambda(P\rho_I)$ such that either $\lambda \in \Lambda(P)$ or $H_{\mathcal{L}I}(\lambda) \in n(P)$.

It is worth mentioning that $\gg$ and $\rightarrow$ are defined only over well-labeled processes. It means that rules (Res), (Par) and (Amb) of Table 10 and the corresponding rules of Table 9, can be applied only when the resulting processes are well-labeled. This guarantees that the well-labeling of processes is preserved, that is the labels of $new(P)$ are fresh. For instance, $R \mid !P \gg R \mid (new(P) \mid !P)$ can be derived by applying rule (Par) to the premise $!P \gg new(P) \mid !P$ provided that both $R \mid (new(P) \mid !P)$ and $R \mid !P$ are well-labeled.

We now show the relation between the reductions of well-labeled processes and those of standard processes. To simplify the proofs we assume that in the inference of a statement $P \equiv Q$ over unlabeled processes the symmetric rules are used directly in place of rule (Symm) (as in $\gg$).

Soundness. We show that any reduction between two well-labeled processes is simulated by a reduction between the corresponding unlabeled processes.

Lemma A.1 *Let P and Q be well-labeled processes. If $P \gg Q$, then $\mathcal{E}(P) \equiv \mathcal{E}(Q)$.*

Proof: It is enough to observe that for any case in Table 10 there exists a corresponding case in Table 3. In the case of (Bang-Bang) we have $P = !P$ and $Q = !P \mid new(P)$. As $\mathcal{E}(new(P)) = \mathcal{E}(P)$, by definition of *new*, we conclude $!\mathcal{E}(P) \equiv !\mathcal{E}(P) \mid \mathcal{E}(new(P))$.

□

[14] We have therefore removed rule (Symm) and introduced the other direction of the rules (Ass), (Res-Par), (Res-Amb), (Nil-Par) and (Nil-Res).

.

Lemma A.2 *Let P be a well-labeled process. If $P \to Q$, then $\mathcal{E}(P) \to \mathcal{E}(Q)$.*

Proof: The proof is straightforward using Lemma A.1 for the case (Cong). $\square$

Completeness. The proof of completeness is more complex. Due to the difference between $\equiv$ and $\gg$, the converse of Lemmas A.1 and A.2 do not hold. Consider for instance the following unlabeled processes

$$P = !R \mid R \mid m[0] \mid n[\mathtt{in}\, m.\, S] \tag{8}$$

$$Q = !R \mid n[\mathtt{in}\, m.\, S'] \mid m[0] \tag{9}$$

We have a reduction $Q \to Q'$ where $Q' = !R \mid m[n[S'] \mid 0]$. Assuming that $S \equiv S'$ we have $P \equiv Q$, and therefore by rule (Cong) we have also $P \to Q'$. We observe that there are no well-labeled versions of P and Q such that $P_{\mathcal{L}} \gg Q_{\mathcal{L}}$ (where $\mathcal{E}(P_{\mathcal{L}}) = P$ and $\mathcal{E}(Q_{\mathcal{L}}) = Q$). The problem is that in $P \equiv Q$ we use: rule (Pref) to derive $\mathtt{in}\, m.\, S \equiv \mathtt{in}\, m.\, S'$, and rule (Bang-Bang) to derive $!R \mid R \equiv !R$. Both steps cannot be simulated by $\gg$ over labeled processes (see Table 10).

We therefore show a weaker property (Lemma A.9): if $P \to Q'$ then there exist well-labeled processes $P_{\mathcal{L}}$ and $Q'_{\mathcal{L}}$, such that $P_{\mathcal{L}} \to Q'_{\mathcal{L}}$, $\mathcal{E}(P_{\mathcal{L}}) = P$ and $\mathcal{E}(Q'_{\mathcal{L}}) \equiv Q'$.

The proof of this property is based on the following steps. We show that, when $P \equiv Q$ and $Q \to Q'$ there exists a special process Q'', such that:

1. $P \gg Q''$ and $Q'' \equiv Q$, where $\gg$ means that only the rules of Table 3 corresponding to those of Table 10 have been used;
2. the derivation $P \gg Q''$ can be simulated in the labeled setting (meaning that there exist well-labeled processes $P_{\mathcal{L}}$ and $Q''_{\mathcal{L}}$, such that $P_{\mathcal{L}} \gg Q''_{\mathcal{L}}$, $\mathcal{E}(P_{\mathcal{L}}) = P$ and $\mathcal{E}(Q''_{\mathcal{L}}) = Q''$);
3. due to the special form of Q'', $Q''_{\mathcal{L}}$ can simulate the transition $Q \to Q'$ (meaning that $Q''_{\mathcal{L}} \to Q'_{\mathcal{L}}$, where $\mathcal{E}(Q'_{\mathcal{L}}) \equiv Q'$).

For instance for the processes (8) and (9) illustrated above we can take

$$Q'' = !R \mid R \mid n[\mathtt{in}\, m.\, S] \mid m[0]. \tag{10}$$

We have $P \gg Q''$ by rules (Comm) and (Par) and we have a transition $Q'' \to Q'''$, where $Q''' = !R \mid R \mid m[n[S] \mid 0]$. Moreover, since $Q' = !R \mid m[n[S'] \mid 0]$ and $S \equiv S'$ we have $Q''' \equiv Q'$ by rules (Bang-Bang), (Pref) and (Par). It is immediate to check that both $P \gg Q''$ and $Q'' \to Q'''$ can be simulated in the labeled setting.

To find out in a systematic way the process which satisfies the properties described above we introduce the following definition.

Let P and Q be processes. We say that a process Q is a *normal form* of a process P iff

- $Q = P = 0$;
- $Q = M.\, Q'$ and $P = M.\, P'$ where $P' \equiv Q'$;
- $Q = Q_1 \mid Q_2$ and $P = P_1 \mid P_2$ where Q_i is a normal form of P_i, for any $i \in \{1, 2\}$;
- $Q = n[Q']$ and $P = n[P']$ where Q' is a normal form of P';
- $Q = (\nu n)\, Q'$ and $P = (\nu n)\, P'$ where Q' is a normal form of P';

- $Q = !Q'$ or $Q = !Q' \mid_{i\in\{1,\dots,n\}} Q'_i$ and $P = !P'$, where $Q' \equiv P'$ and $Q' \equiv Q'_i \equiv P'$ for any $i \in \{1,\dots,n\}$.

For instance the process (10) is a normal form of the process (9).

We give below some easy properties about the normal form.

Proposition A.3 *Let P, Q be processes such that Q is a normal form of P. We have $P \equiv Q$.*

Proof: The proof proceeds by induction on the structure of P using the definition of normal form and the rules of Table 3. The most interesting case is when $P = !P'$ and either $Q = !Q'$ or $Q = !Q' \mid_{i\in\{1,\dots,n\}} Q'_i$, where $Q' \equiv P'$ and $Q'_i \equiv P'$ for $i \in \{1,\dots,n\}$. In the former case $P \equiv Q$ follows immediately by rule (Bang). In the latter case we have $!P' \equiv !Q'$ by rule (Bang). Also, by rules (Bang-Bang), (Par) and (Trans) $!Q' \equiv !Q' \mid_{i\in\{1,\dots,n\}} Q'_i$. Thus, by rule (Trans) we have $P \equiv Q$.

□

Proposition A.4 *Let P_1, P_2 and P_3 be processes. If P_1 is a normal form of P_2 and P_2 is a normal form of P_3, then P_1 is a normal form of P_3.*

Proof: All the cases are easy using the definition of normal form except from the case when $P_3 = !Q$. By definition we have either $P_2 = !P$ or $P_2 = !P \mid_{i\in\{1,\dots,n\}} Q_i$, where $Q \equiv P \equiv Q_i$, for any $i \in \{1,\dots,n\}$. In the former case, we have $P_1 = !R$ or $P_1 = !R \mid_{i\in\{1,\dots,k\}} R_i$, where $P \equiv R \equiv R_i$, for any $i \in \{1,\dots,k\}$. Since $Q \equiv P$ we have also $Q \equiv R \equiv R_i$, for any $i \in \{1,\dots,k\}$. Consequently, P_1 is a normal form of P_3. In the latter case, we have $P_1 = P_{1,1} \mid P_{2,1}$ where $P_{1,1}$ is a normal form of $!P$ and $P_{2,1}$ is a normal form of $\mid_{i\in\{1,\dots,n\}} Q_i$. It means that $P_{1,1} = !R$ or $P_{1,1} = !R \mid_{i\in\{1,\dots,k\}} R_i$, where $P \equiv R \equiv R_i$, for any $i \in \{1,\dots,k\}$. Moreover, $P_{2,1} = \mid_{i\in\{1,\dots,n\}} S_i$, where S_i is a normal form of Q_i. By Proposition A.3 we have $S_i \equiv Q_i$ for any $i \in \{1,\dots,k\}$. Assume that $P_1 = !R \mid_{i\in\{1,\dots,n\}} S_i$. Since $Q \equiv P \equiv Q_i$, $S_i \equiv Q_i$ and $P \equiv R$, P_1 is a normal form of P_3. Assume that $P_1 = !R \mid_{i\in\{1,\dots,k\}} R_i \mid\mid_{i\in\{1,\dots,n\}} S_i$. Since $Q \equiv P \equiv R \equiv R_i$ and $S_i \equiv Q_i \equiv P$, then P_1 is a normal form of P_3.

□

We show the main property of normal forms we have discussed above: if $P \equiv Q$ then there exists a normal form Q' of Q such that $P \gg Q'$ and $Q' \equiv Q$.

Lemma A.5 *Let P, Q be processes such that $P \equiv Q$. There exists a process Q', which is a normal form of Q, such that $P \gg Q'$ and $Q' \equiv Q$.*

Proof: We notice that, by Proposition A.3, when Q' is a normal form of Q, we have also $Q' \equiv Q$. Therefore, it is enough to find out a process Q', which is a normal form of Q. The proof proceeds by induction on the depth of the inference of $P \equiv Q$.

- The cases of (Refl), (Comm), (Ass), (Res-Com), (Res-Par), (Res-Amb), (Nil-Par) and (Nil-Res) are easy. They can be solved by taking $Q' = Q$, as $P \gg Q'$ follows from $P \equiv Q$ (by applying the same rule of Table 10).
- The cases of (Res), (Par) and (Amb) are similar and follow by applying the induction hypothesis; as an example we show (Amb). It means that $P = n[R']$ and $Q = n[R]$, where $R' \equiv R$. Since $R' \equiv R$, by induction hypothesis there exists R'', such that R'' is a normal form of R and $R' \gg R''$. We take $Q' = n[R'']$. We have $P \gg Q'$ by applying rule (Amb) to the premise $R' \gg R''$. Moreover, since R'' is a normal form of R, then Q' is a normal form of Q.

- In case (Bang) we have $P = !R'$ and $Q = !R$, where $R' \equiv R$. Taking $Q' = P$ we immediately have $P \gg Q'$ by rule (Refl). Moreover, since $R' \equiv R$ (and conversely $R \equiv R'$) then Q' is a normal form of Q.
- In case (Pref) we have $P = M.R'$ and $Q = M.R$, where $R' \equiv R$. Taking $Q' = P$ we have $P \gg Q'$ by rule (Refl). Since $R' \equiv R$, then Q' is a normal form of Q.
- In case (Bang-Bang) there are two possibilities depending on the way the rule is applied. Therefore, either $P = !R \mid R$ and $Q = !R$ or $P = !R$ and $Q = !R \mid R$. In the latter case we take $Q' = Q$ and we have $P \gg Q'$ by rule (Bang-Bang). In the former case we take $Q' = P$ and we have $P \gg Q'$ by rule (Refl). Moreover, Q' is a normal form of Q using $R \equiv R$.
- In case (Trans) we have $P \equiv Q_1$ and $Q_1 \equiv Q$. By induction hypothesis there exist R_1, R_2 such that: (i) $P \gg R_1$ and R_1 is a normal form of Q_1; (ii) $Q_1 \gg R_2$ and R_2 is a normal form of Q.

  This case is rather complex. The crux of the proof consists of showing that, since R_1 is a normal form of Q_1 (and thus by Proposition A.3 $R_1 \equiv Q_1$) and $Q_1 \gg R_2$, then there exists a process Q', which is a normal form of R_2, such that $R_1 \gg Q'$ (and by Proposition A.3 $Q' \equiv R_2$). To prove this property we proceed by induction on the depth of the inference of $Q_1 \gg R_2$. The case (Refl) is obvious; we show the other cases below.

  - The cases of (Comm) and (Ass) are similar; as an example we show (Comm). We have $Q_1 = S_1 \mid S_2$ and $R_2 = S_2 \mid S_1$. Since R_1 is a normal form of Q_1 it means that $R_1 = S'_1 \mid S'_2$, where S'_i is a normal form of S_i for any $i \in \{1, 2\}$. We take $Q' = S'_2 \mid S'_1$ and we have $R_1 \gg Q'$ by rule (Comm). Moreover, Q' is a normal form of R_2, since S'_i is a normal form of S_i for any $i \in \{1, 2\}$.
  - The cases of (Res), (Par) and (Amb) are similar; as an example we show (Amb). We have $Q_1 = n[S]$ and $R_2 = n[S']$ where $S \gg S'$. Since R_1 is a normal form of Q_1 it means that $R_1 = n[S'']$, where S'' is a normal form of S. As S'' is a normal form of S and $S \gg S'$, by induction hypothesis there exists S''', which is a normal form of S', such that $S'' \gg S'''$. We take $Q' = n[S''']$. As S''' is a normal form of S', then Q' is a normal form of R_2. Moreover, we have $R_1 \gg Q'$ by applying rule (Amb) to the premise $S'' \gg S'''$.
  - The cases of (Res-Com), (Res-Par), (Res-Amb), (Nil-Par) and (Nil-Res) are similar; as an example we show (Res-Par). There are two cases: either $Q_1 = (\nu n)\,(S_1 \mid S_2)$ and $R_2 = S_1 \mid (\nu n)\,S_2$ or the converse. We show only the former case, the other is analogous.

    Since R_1 is a normal form of Q_1 it means that $R_1 = (\nu n)\,S'_1 \mid S'_2$, where S'_i is a normal form of S_i for any $i \in \{1, 2\}$. Taking $Q' = S'_1 \mid (\nu n)\,S'_2$ we have that Q' is a normal form of R_2. Moreover, we have $R_1 \gg Q'$ by applying rule (Res-Par).
  - In case (Bang-Bang) we have $Q_1 = !S$ and $R_2 = !S \mid S$. Since R_1 is a normal form of Q_1 it means that either $R_1 = !S'$ or $R_1 = !S' \mid_{i \in \{1,\ldots,n\}} S'_i$ where $S \equiv S'$ (and conversely $S' \equiv S$) and $S \equiv S'_i$ (and conversely $S'_i \equiv S$), for any $i \in \{1, \ldots, n\}$. We show only the former case; the other is analogous.

    We observe that R_1 is a normal form of Q_1, and $P \equiv Q_1$, and $P \gg R_1$, where $R_1 = !S'$ and $Q_1 = !S$. It means that rule (Bang) is applied in $P \equiv Q_1$ to the premise $S' \equiv S$ (see the case (Bang) above). Therefore, by applying the induction hypothesis to $S' \equiv S$, there exists S'', which is a normal form of S, such that $S' \gg S''$.

We take $Q' = !S' \mid S''$. By applying rule (Bang-Bang) we have $R_1 \gg !S' \mid S'$. Moreover, by applying rule (Par) to the premise $S' \gg S''$ we obtain $!S' \mid S' \gg !S' \mid S''$. Hence, by rule (Trans) we have $R_1 \gg Q'$. We conclude by observing that Q' is a normal form of R_2 as $S \equiv S'$ and S'' is a normal form of S.

- In case (Trans) we have $Q_1 \gg S_1$ and $S_1 \gg R_2$. As R_1 is a normal form of Q_1, then by induction hypothesis there exists a process Q'', which is a normal form of S_1, such that $R_1 \gg Q''$. Since Q'' is a normal form of S_1 and $S_1 \gg R_2$, then by induction hypothesis there exists a process Q', which is a normal form of R_2, such that $Q'' \gg Q'$. We conclude by observing that by applying rule (Trans) to the premises $R_1 \gg Q''$ and $Q'' \gg Q'$, we obtain $R_1 \gg Q'$.

Using the property above[15] we now conclude the case (Trans). Since Q' is a normal form of R_2 and R_2 is a normal form of Q (condition (ii)), we have by Proposition A.4 that Q' is a normal form of Q. Moreover, $P \gg Q'$ follows from $P \gg R_1$ (condition (i)) and $R_1 \gg Q'$.

$\square$

We present now two auxiliary properties of the relation $\gg$ and of the reduction relation over well-labeled processes. They show that the new labels introduced in a process by $\gg$ or by a reduction can be properly re-indexed. This is possible because new labels can be introduced only by rule (Bang-Bang) of Table 10 by means of *new*.

Proposition A.6 *Let P and Q be well-labeled processes such that $P \gg Q$. We have $fn(P) = fn(Q)$, and for each re-indexing of labels ρ_I, such that $dom(\rho_I) = \Lambda(Q) \setminus \Lambda(P)$, and $Q\rho_I$ is well-labeled, we have also $P \gg Q\rho_I$.*

Proof: The proof proceeds by induction on the depth of the inference of $P \gg Q$. We observe that in any rule of Table 10, $P \gg Q$ implies $fn(P) = fn(Q)$. Moreover, in any rule of Table 10, $P \gg Q$ implies $\Lambda(P) = \Lambda(Q)$, except from rules (Bang-Bang), (Nil-Res) and rules (Res), (Par), (Amb), (Trans).

- Suppose that rule (Nil-Res) has been applied. We have $P = 0$ and $Q = (\nu n_\lambda)0$ or vice-versa. The latter case is immediate, in the former case we have $\Lambda(Q) \setminus \Lambda(P) = \{\lambda\}$. Hence, for any re-indexing of labels ρ_I such that $Q\rho_I$ is well-labeled, we have $Q\rho_I = (\nu n_{\rho_I(\lambda)})\,0$. We conclude as follows by $P \gg Q\rho_I$ by rule (Nil-Res).
- Suppose that rule (Bang-Bang) has been applied. We have $P = !P_1$ and $Q = !P_1 \mid new(P_1)$. It means that $new(P_1) = P_1{\rho'}_I$ for a re-indexing of labels such that: $dom({\rho'}_I) = \Lambda(P_1)$; $P_1{\rho'}_I$ is well-labeled; there is no $\lambda \in \Lambda(P_1{\rho'}_I)$ such that either $\lambda \in \Lambda(P_1)$ or $H_{\mathcal{L}_I}(\lambda) \in n(P_1)$. These conditions ensure that $new(P_1) \mid !P_1$ is well-labeled, and therefore that $\Lambda(new(P_1)) \cap \Lambda(!P_1) = \emptyset$. Let ρ_I be a re-indexing of labels such that $dom(\rho_I) = \Lambda(Q) \setminus \Lambda(P)$ and $Q\rho_I = (new(P_1) \mid !P_1)\rho_I$ is well-labeled. Since $\Lambda(new(P_1)) \cap \Lambda(!P_1) = \emptyset$, we have $dom(\rho_I) = \Lambda(new(P_1))$ and $(new(P_1) \mid !P_1)\rho_I = new(P_1)\rho_I \mid !P_1$. Since $new(P_1)\rho_I \mid !P_1$ is well-labeled, also $P_1{\rho'}_I\rho_I$. Thus, we can apply rule (Bang-Bang) to conclude $!P_1 \gg P_1{\rho'}_I\rho_I \mid !P_1$.

[15]There exists a process Q', which is a normal form of R_2, such that $R_1 \gg Q'$.

- The cases of rules (Res), (Par), (Amb) and (Trans) are similar and follow by induction hypothesis using the well-labeling of P. We give as an example (Par) and (Res).

  Assume that $P = P_1 \mid P_2$ and $Q = Q_1 \mid P_2$, where $P_1 \gg Q_1$. Let ρ_I be a re-indexing of labels, such that $dom(\rho_I) = \Lambda(Q) \setminus \Lambda(P)$, and $Q\rho_I$ is well-labeled. Since P and Q are well-labeled, then $\Lambda(P_1) \cap \Lambda(P_2) = \emptyset$ and $\Lambda(Q_1) \cap \Lambda(P_2) = \emptyset$. Therefore, we have $\Lambda(Q) \setminus \Lambda(P) = \Lambda(Q_1) \setminus \Lambda(P_1)$ and $Q\rho_I = Q_1\rho_I \mid P_2$. We observe that $Q_1\rho_I$ is well-labeled, since $Q\rho_I$ is well-labeled. Thus, by induction hypothesis $P_1 \gg Q_1\rho_I$. We conclude by applying rule (Par) to derive $P_1 \mid P_2 \gg Q_1\rho_I \mid P_2$.

  Assume that $P = (\nu n_\lambda)\, P_1$ and $Q = (\nu n_\lambda)\, Q_1$, where $P_1 \gg Q_1$. Let ρ_I be a re-indexing of labels, such that $dom(\rho_I) = \Lambda(Q) \setminus \Lambda(P)$, and $Q\rho_I$ is well-labeled. Since Q is well-labeled, then $\lambda \notin \Lambda(Q_1)$, $H_{\mathcal{L}_I}(\lambda) \notin n(Q_1)$, and there is no $\mu \in \Lambda(Q_1)$ such that $H_{\mathcal{L}_I}(\mu) = n$. Therefore, we have $\lambda \notin \Lambda(Q) \setminus \Lambda(P)$, and consequently $Q\rho_I = (\nu n_\lambda)\,(Q_1\rho_I)$. We observe that $Q_1\rho_I$ is well-labeled, as $Q\rho_I$ is well-labeled. Hence, by induction hypothesis $P_1 \gg Q_1\rho_I$. We conclude by applying rule (Res) to derive $(\nu n_\lambda)\, P_1 \gg (\nu n_\lambda)\,(Q_1\rho_I)$.

□

Proposition A.7 *Let P and Q be well-labeled processes such that $P \to Q$. We have $fn(Q) \subseteq fn(P)$, and for each re-indexing of labels ρ_I, such that $dom(\rho_I) = \Lambda(Q) \setminus \Lambda(P)$, and $Q\rho_I$ is well-labeled, we have also $P \to Q\rho_I$.*

Proof: The proof proceeds by induction on the depth of the inference of $P \to Q$. The cases of (In), (Out), and (Open) are immediate given that $\Lambda(Q) \subseteq \Lambda(P)$. The case of rule (Cong) follows by Proposition A.6. The cases of (Par), (Amb) and (Res) can be proved by induction following a reasoning similar to that used in the corresponding cases of Proposition A.6.

□

The following lemma shows that the converse of Lemma A.1 holds for unlabeled processes related by $\gg$.

Lemma A.8 *Let P be a well-labeled process. If $\mathcal{E}(P) \gg Q$, then there exists a well-labeled process Q', such that $P \gg Q'$ and $\mathcal{E}(Q') = Q$.*

Proof: We proceed by induction on the derivation of $\mathcal{E}(P) \gg Q$ using the fact that for any rule of Table 3, which could have been applied to derive $\gg$, there exists a corresponding case in Table 10. We discuss the most interesting cases, the others are trivial.

- In case (Bang-Bang) we have $\mathcal{E}(P) = !\mathcal{E}(P_1)$ and $Q = !\mathcal{E}(P_1) \mid \mathcal{E}(P_1)$. Let $Q' = !P_1 \mid new(P_1)$, which is (by definition of *new*) well-labeled. By rule (Bang-Bang) we have $!P_1 \gg !P_1 \mid new(P_1)$ and $\mathcal{E}(Q') = !\mathcal{E}(P_1) \mid \mathcal{E}(new(P_1)) = !\mathcal{E}(P_1) \mid \mathcal{E}(P_1) = Q$.
- In cases (Res),(Par), (Amb) we apply the induction hypothesis using Proposition A.6 to find out the well-labeled process Q'. We show as an example the cases of (Par) and (Res).

  Assume that $\mathcal{E}(P) \gg Q$ has been derived by rule (Par). It means that $P = P_1 \mid R$ and $Q = Q_1 \mid \mathcal{E}(R)$, where $\mathcal{E}(P_1) \gg Q_1$. By induction hypothesis, there exists a well-labeled process Q'_1 such that $\mathcal{E}(Q'_1) = Q_1$ and $P_1 \gg Q'_1$. Using Proposition A.6 we derive that $fn(Q'_1) = fn(P_1)$. Moreover, since $P_1 \mid R$ is well-labeled we have: (i) $\Lambda(P_1) \cap \Lambda(R) = \emptyset$; (ii) for each $\lambda \in \Lambda(P_1)$, $H_{\mathcal{L}-I}(\lambda) \notin n(R)$; conversely (iii) for each $\lambda \in \Lambda(R)$, $H_{\mathcal{L}_I}(\lambda) \notin n(P_1)$.

We now use the fact that the labels $\Lambda(Q_1') \setminus \Lambda(P_1)$ can be re-indexed. Therefore, let ρ_I be a re-indexing of labels, such that $dom(\rho_I) = \Lambda(Q_1') \setminus \Lambda(P_1)$, $Q_1'\rho_I$ is well-labeled, $\Lambda(Q_1'\rho_I) \cap \Lambda(R) = \emptyset$ and, for each $\lambda \in \Lambda(Q_1'\rho_I)$, $H_{\mathcal{L}_I}(\lambda) \notin n(R)$. As $Q_1'\rho_I$ is well-labeled, then by Proposition A.6, we obtain $P_1 \gg Q_1'\rho_I$.

We now observe that $fn(Q_1') = fn(P_1)$ and $fn(Q_1') = fn(Q_1'\rho_I)$ and that the bound names of $Q_1'\rho_I$ can be properly α-converted. By condition (iii) above we derive that $H_{\mathcal{L}_I}(\lambda) \notin n(Q_1'\rho_I)$ for any $\lambda \in \Lambda(R)$. Moreover, ρ_I has been chosen to have $\Lambda(Q_1'\rho_I) \cap \Lambda(R) = \emptyset$ and, for each $\lambda \in \Lambda(Q_1'\rho_I)$, $H_{\mathcal{L}_I}(\lambda) \notin n(R)$. Therefore, $Q_1'\rho_I \mid R$ is a well-labeled process.

Let $Q' = Q_1'\rho_I \mid R$. Since $P_1 \gg Q_1'\rho_I$, then by rule (Par) of Table 10 we have $P \gg Q'$. Moreover, since $\mathcal{E}(Q_1'\rho_I) = \mathcal{E}(Q_1') = Q_1$ we conclude that $\mathcal{E}(Q_1'\rho_I \mid R) = \mathcal{E}(Q_1') \mid \mathcal{E}(R) = Q_1 \mid \mathcal{E}(R) = Q$.

Assume that $\mathcal{E}(P) \gg Q$ has been derived by rule (Res). It means that $P = (\nu n_\lambda)\, P_1$ and $Q = (\nu n)\, Q_1$, where $\mathcal{E}(P_1) \gg Q_1$. By induction hypothesis, there exists a well-labeled process Q_1' such that $\mathcal{E}(Q_1') = Q_1$ and $P_1 \gg Q_1'$. Using Proposition A.6 we derive that $fn(Q_1') = fn(P_1)$. Moreover, since $(\nu n_\lambda)\, P_1$ is well-labeled we have: (i) $\lambda \notin \Lambda(P_1)$; (ii) for each $\mu \in \Lambda(P_1)$, $H_{\mathcal{L}_I}(\mu) \neq n$; conversely (iii) $H_{\mathcal{L}_I}(\lambda) \notin n(P_1)$.

We now use the fact that the labels $\Lambda(Q_1') \setminus \Lambda(P_1)$ can be re-indexed. Therefore, let ρ_I be a re-indexing of labels, such that $dom(\rho_I) = \Lambda(Q_1') \setminus \Lambda(P_1)$, $Q_1'\rho_I$ is well-labeled, $\lambda \notin \Lambda(Q_1'\rho_I)$ and, for each $\mu \in \Lambda(Q_1'\rho_I)$, $H_{\mathcal{L}_I}(\mu) \neq n$. As $Q_1'\rho_I$ is well-labeled, then by Proposition A.6, we obtain $P_1 \gg Q_1'\rho_I$.

We now observe that $fn(Q_1') = fn(P_1)$ and $fn(Q_1') = fn(Q_1'\rho_I)$ and that the bound names of $Q_1'\rho_I$ can be properly α-converted. Since for each $\mu \in \Lambda(Q_1'\rho_I)$, $H_{\mathcal{L}_I}(\mu) \neq n$, and by conditions (i) and (iii) above, we derive that $(\nu n_\lambda)\, Q_1'\rho_I$ is well-labeled.

Let $Q' = (\nu n_\lambda)\, Q_1'\rho_I$. Since $P_1 \gg Q_1'\rho_I$, then by rule (Res) of Table 10 we have $P \gg Q'$. Moreover, since $\mathcal{E}(Q_1'\rho_I) = \mathcal{E}(Q_1') = Q_1$ we conclude that $\mathcal{E}(Q') = (\nu n)\, Q_1 = Q$.

□

Using Lemmas A.5 and A.8 and the shape of normal forms we can now prove the main result of completeness.

Lemma A.9 *Let P be a well-labeled process. If $\mathcal{E}(P) \to Q$, then there exists a well-labeled process Q' such that $\mathcal{E}(Q') \equiv Q$ and $P \to Q'$.*

Proof: We prove a more general result: if $\mathcal{E}(P) \equiv P_1$ and $P_1 \to Q$, then there exists a well-labeled process Q' such that $\mathcal{E}(Q') \equiv Q$ and $P \to Q'$. For this we proceed by induction on the depth of the inference of $P_1 \to Q$.

- The cases of (In), (Out), and (Open) are similar; as an example we show (In). If $P_1 \to Q$ has been obtained by rule (In), it means that $P_1 = n[\mathtt{in}\, m.\, R_1 \mid R_2] \mid m[S]$ and $Q = m[n[R_1 \mid R_2] \mid S]$.

  Since $\mathcal{E}(P) \equiv P_1$, then by Lemma A.5 there exists a process P_1', which is a normal form of P_1, such that $\mathcal{E}(P) \gg P_1'$ and $P_1' \equiv P_1$.

  We now apply Lemma A.8. As $\mathcal{E}(P) \gg P_1'$, then there exists a well-labeled process P_1'' such that $P \gg P_1''$ and $\mathcal{E}(P_1'') = P_1'$. Since P_1' is a normal form of P_1, then it must be the case that
$$P_1'' = n_\lambda[\mathtt{in}\, m_\gamma.\, R_1' \mid R_2'] \mid m_\mu[S']$$

$n_\lambda[\mathtt{in}\, m_\gamma.\, P \mid Q] \mid m_\mu[R] \to m_\mu[n_\lambda[P \mid Q] \mid R]$	(In)
$m_\mu[n_\lambda[\mathtt{out}\, m_\gamma.\, P \mid Q] \mid R] \to n_\lambda[P \mid Q] \mid m_\mu[R]$	(Out)
$\mathtt{open}\, n_\mu.\, P \mid n_\lambda[Q] \to P \mid Q$	(Open)
$P \to Q \Rightarrow (\nu n_\lambda)\, P \to (\nu n_\lambda)\, Q$	(Res)
$P \to Q \Rightarrow P \mid R \to Q \mid R$	(Par)
$P \to Q \Rightarrow n_\lambda[P] \to n_\lambda[Q]$	(Amb)
$(P' \to Q',\ P \gg P',\ Q' \gg Q) \Rightarrow P \to Q$	(Cong)

Table 9: Reductions for well-labeled processes

where $\mathcal{E}(R_1') \equiv R_1$, $\mathcal{E}(R_2')$ is a normal form R_2 and $\mathcal{E}(S')$ is a normal form of S.

By applying rule (In) we have a reduction $P_1'' \to Q'$, where

$$Q' = m_\mu[n_\lambda[R_1' \mid R_2'] \mid S'].$$

Moreover, since $P \gg P_1''$ we have by rule (Cong) $P \to Q'$.

We conclude by observing that $\mathcal{E}(R_2') \equiv R_2$ and $\mathcal{E}(S') \equiv S$ (using Proposition A.3). Given that also $\mathcal{E}(R_1') \equiv R_1$, $\mathcal{E}(Q') \equiv Q$ follows by applying rules (Par), (Amb) and (Trans).

- The cases of (Par), (Amb) and (Res) are similar; they follow by applying the induction hypothesis and by using Proposition A.7 to find out the well-labeled process Q' (similarly as in the proof of Lemma A.8). We show as an example the case (Par).

  Assume that $P_1 \to Q$ has been obtained by rule (Par). It means that $P_1 = Q_1 \mid R$ and $Q = Q_2 \mid R$, where $Q_1 \to Q_2$.

  Since $\mathcal{E}(P) \equiv P_1$, then by Lemma A.5 there exists a process P_1', which is a normal form of P_1, such that $\mathcal{E}(P) \gg P_1'$ and $P_1' \equiv P_1$.

  We now apply Lemma A.8. As $\mathcal{E}(P) \gg P_1'$, then there exists a well-labeled process P_1'' such that $P \gg P_1''$ and $\mathcal{E}(P_1'') = P_1'$. Since P_1' is a normal form of P_1, then it must be the case that $P_1'' = Q_1' \mid R'$, where $\mathcal{E}(Q_1')$ is a normal form of Q_1 and $\mathcal{E}(R')$ is a normal form of R. By Proposition A.3 we have that $\mathcal{E}(Q_1') \equiv Q_1$ and $\mathcal{E}(R') \equiv R$. Since $\mathcal{E}(Q_1') \equiv Q_1$ and $Q_1 \to Q_2$, by induction hypothesis there exists a reduction $Q_1' \to Q_2'$ such that $\mathcal{E}(Q_2') \equiv Q_2$.

  We now use Proposition A.7 to find out a re-indexing of labels ρ_I such that $Q_1' \to Q_2'\rho_I$ and $Q' = Q_2'\rho_I \mid R'$ is well-labeled (the reasoning follows an argument similar to that applied in the proof of Lemma A.8).

  As Q' is well-labeled, then we derive $P_1'' \to Q'$ by applying rule (Par) to the premise $Q_1' \to Q_2'\rho_I$. Since $P \gg P_1''$ we have also $P \to Q'$ by rule (Cong).

It remains to show that $\mathcal{E}(Q') \equiv Q$. We recall that $Q = Q_2 \mid R$ and $Q' = Q'_2\rho_I \mid R'$, where $\mathcal{E}(Q'_2) \equiv Q_2$ and $\mathcal{E}(R') \equiv R$. Given that $\mathcal{E}(Q'_2) = \mathcal{E}(Q'_2\rho_I)$, $\mathcal{E}(Q') \equiv Q$ follows therefore by rules (Par) and (Trans).

- If $P_1 \to Q$ has been obtained by rule (Cong) it means that $P_1 \equiv P_2$, $P_2 \to P_3$ and $P_3 \equiv Q$. As $\mathcal{E}(P) \equiv P_1$ and $P_1 \equiv P_2$ we have by rule (Trans) $\mathcal{E}(P) \equiv P_2$. Since $\mathcal{E}(P) \equiv P_2$ and $P_2 \to P_3$, then by induction hypothesis there exists $P \to Q'$ such that $\mathcal{E}(Q') \equiv P_3$. We conclude by observing that $\mathcal{E}(Q') \equiv Q$ follows by applying rule (Trans) to the premises $\mathcal{E}(Q') \equiv P_3$ and $P_3 \equiv Q$.

□

A.2 Relation between the normal semantics and the reductions of labeled processes

We start presenting the basic properties of the normalisation function δ (Table 4). The following proposition shows that α-convertible processes are represented by the same state. We recall that α-conversion over labeled processes can change a bound name but not its label.

Proposition A.10 (α-conversion) *Let P and Q be two well-labeled processes which are α-convertible. For any $a \in \mathcal{A}$, which is fresh for P and Q, we have $\delta\ {}^{a}P = \delta\ {}^{a}Q$.*

Proof: The main observation is the following: when $P = (\nu n_\lambda)\, P_1$ and $Q = (\nu k_\lambda)\, P_1[k/n]$, such that $k \notin fn(P_1)$, we have by rule **DRes** $\delta\ {}^{a}P = \delta\ {}^{a}(P_1[H_{\mathcal{L}_I}(\lambda)/n]) = \delta\ {}^{a}Q = \delta\ {}^{a}(P_1[k/n][H_{\mathcal{L}_I}(\lambda)/k])$.
□

We now discuss the relation between the (free and bound) names and the labels of a process and those of the corresponding state obtained via δ. To formalise this relation it is necessary to know precisely which restrictions are removed via δ. We therefore introduce the following concepts which use a special kind of contexts. A context C is a process expression with a single occurrence of a hole [], such that the hole does not appear underneath the scope of a prefix or of a bang. As usual we denote by $C[P]$ the process obtained by filling the hole of C with the process P.

Let P be a labeled process. If $P = C[(\nu n_\lambda)Q]$ for some context C, then we say that (νn_λ) is an *unguarded* restriction of P; if also $n \notin fn(Q)$ we say that (νn_λ) is an *unguarded and unnecessary* restriction of P.

For instance, the restriction (νn_λ) is unguarded and the restriction (νm_γ) is not unguarded in the following process $P = a[(\nu n_\lambda)\, !(\nu m_\gamma)\, Q]$.

The unguarded restrictions of a process are important, as they are removed by the normalisation function δ. For instance, we have for the process P above

$$\delta\ {}^{@}P = (\{\ {}_{a}^{@}\}, \{\ {}^{a}!(\nu m_\gamma)\, Q[H_{\mathcal{L}_I}(\lambda)/n]\}).$$

The difference between the unguarded and the unguarded and unnecessary restrictions of a process is the following: if (νn_λ) is an unguarded and unnecessary restriction of a process P, then $H_{\mathcal{L}_I}(\lambda)$ does not necessarily appear in the state modeling P. For instance, assume that the process P above is well-labeled, that is $H_{\mathcal{L}_I}(\lambda) \notin n(Q) \cup n(a)$. The name $H_{\mathcal{L}_I}(\lambda)$ appears in the state $\delta\ {}^{@}P$ only when $n \in fn(Q)$.

These intuitive ideas are stated by the propositions below. In the following, we use $\mathcal{U}(P) = \{n_\lambda \mid (\nu n_\lambda)$ is an unguarded restriction of $P\}$ and $\mathcal{U}_u(P) = \{n_\lambda \mid (\nu n_\lambda)$ is an unguarded and unnecessary restriction of $P\}$.

Proposition A.11 *Let P be a well-labeled process and let $a \in \mathcal{A}$ which is fresh for P. We have*

1. *if (νn_λ) and (νm_μ) are two distinct unguarded restrictions of P, then $\lambda \neq \mu$;*
2. *for any $n_\lambda \in \mathcal{U}(P)$, $H_{\mathcal{L}_I}(\lambda) \notin n(P)$;*
3. *$H_{\mathcal{L}_I}(\lambda) \neq H_{\mathcal{L}_I}(\mu)$ for any $n_\lambda, m_\mu \in \mathcal{U}(P)$.*

Proof: The conditions follow straightforwardly from the definition of well-labeled process (Definition 4.2). $\square$

Proposition A.12 *Let P be a well-labeled process and $a \in \mathcal{A}$ which is fresh for P. We have*

- *$\Lambda(\delta\ ^a P) \setminus \Lambda(a) = \Lambda(P) \setminus \{\lambda \mid n_\lambda \in \mathcal{U}(P)\}$;*
- *$n(\delta\ ^a P) \setminus n(a) = fn(P) \cup (bn(P) \setminus \{n \mid n_\lambda \in \mathcal{U}(P)\}) \ \cup\ \{H_{\mathcal{L}_I}(\lambda) \mid n_\lambda \in (\mathcal{U}(P) \setminus \mathcal{U}_u(P))\}$.*

Proof: The requirements on $\Lambda(\delta\ ^a P)$ and $n(\delta\ ^a P)$ can be proved by induction on the structure of P using using Proposition A.11. The main observation is that, by definition of δ, only the unguarded restrictions are removed (see rules **DBang** and **DPref**). In case **DRes**, we have for $P = (\nu n_\lambda)\, Q$

$$\delta\ ^a P = \delta\ ^a (Q[H_{\mathcal{L}_I}(\lambda)/n]).$$

This shows that the label λ is removed and the name n is replaced by $H_{\mathcal{L}_I}(\lambda)$. We recall that, by Proposition A.11: for any $n_\lambda \in \mathcal{U}(P)$, $H_{\mathcal{L}_I}(\lambda) \notin n(P)$ and, there is no other object in P with label λ. Therefore, $H_{\mathcal{L}_I}(\lambda) \in n(\delta\ ^a P)$ only when $n \in fn(Q)$, that is $(\nu n_\lambda) \in \mathcal{U}(P) \setminus \mathcal{U}_u(P)$.
$\square$

The following proposition is needed in the proof of completeness (Lemma A.20); it says that the state representing a well-labeled process is well-labeled provided that the root a is fresh for P. We recall that a state $S \in \mathcal{S}$ is *well-labeled* if: (i) for each $\lambda \in \Lambda(S)$, $H_{\mathcal{L}_I}(\lambda) \notin n(S)$; (ii) for any label $\lambda \in \Lambda(S)$ there is at most one object labeled by λ.

Proposition A.13 *Let P be a well-labeled process and let $a \in \mathcal{A}$, such that a is fresh for P. We have that $\delta\ ^a P$ is a well-labeled state with root a.*

Proof: Straightforward by induction on the structure of P using Propositions A.11 and A.12.
$\square$

The converse of Proposition A.13 does not hold. Consider, for instance, the following not well-labeled process

$$P = (\nu n_\lambda)\, m_\lambda[0] \tag{11}$$

We have $\delta\ ^@ P = (\ _{\{m_\lambda}{}^@\}, \emptyset)$ which is obviously well-labeled.

The anomaly in process (11) is that (νn_λ) is an unguarded and unnecessary restriction; therefore the name $H_{\mathcal{L}_I}(\lambda)$, that is used to replace the bound name n, does not appear in the state representing P (see Proposition A.12). By contrast, the clash between the two occurrences of label λ is necessarily reflected into the corresponding state, when the bound name appears in the process. Consider, for instance, the following not well-labeled process

$$Q = (\nu n_\lambda)\, m_\lambda[\texttt{out}\, n] \tag{12}$$

We have $\delta\ {}^{@}Q = (\ \{{}_{m_\lambda}{}^{@}\}, \{\ {}^{m_\lambda}\mathtt{out}\,\hat{n}\})$ where $H_{\mathcal{L}_I}(\lambda) = \hat{n}$. We observe that $\delta\ {}^{@}Q$ is not well-labeled since $\hat{n} \in n(\delta\ {}^{@}Q)$ and $\lambda \in \Lambda(\delta\ {}^{@}Q)$.

There is a main difference between the processes (11) and (12) above. In case (11) the process can be properly rearranged and a well-labeled process P' can be obtained, such that $\delta\ {}^{@}P = \delta\ {}^{@}P'$ and $P \gg P'$. For instance, taking $P' = m_\lambda[0]$, it is immediate to check that $\delta\ {}^{@}P = \delta\ {}^{@}P'$ and $P \gg P'$, since $n \notin fn(m_\lambda[0])$ (reflecting the idea that this restriction is unnecessary). For the process (12) instead there is no way to modify the labels using $\gg$.

The idea explained for the processes P and P' above is useful in the proof of soundness (Lemma A.17). We therefore formalise it by introducing a relation $\triangleright$ and by showing that: when $P \triangleright P'$, we have $\delta\ {}^{a}P = \delta\ {}^{a}P'$ and $P \gg P'$ (and vice-versa $P' \gg P$). The intuitive idea behind $\triangleright$ is that P' is obtained from P by eliminating all the unguarded and unnecessary restrictions. We define the relation $\triangleright$ over labelled processes inductively as follows:

1. $0 \triangleright 0$, $!P \triangleright !P$, $M_\lambda.P \triangleright M_\lambda.P$;
2. $Q \mid P \triangleright Q' \mid P'$ provided that $Q \triangleright Q'$ and $P \triangleright P'$;
3. $a[Q] \triangleright a[Q']$ provided that $Q \triangleright Q'$;
4. $(\nu n_\lambda)\ Q \triangleright (\nu n_\lambda)\ Q'$ provided that $Q \triangleright Q'$ and $n \in fn(Q)$;
5. $(\nu n_\lambda)\ Q \triangleright Q'$ provided that $Q \triangleright Q'$ and $n \notin fn(Q)$.

Notice that by condition 5. we have $\mathcal{U}_u(P') = \emptyset$ when $P \triangleright P'$. Moreover, we have immediately $fn(P) = fn(P')$ and $\Lambda(P') \subseteq \Lambda(P)$.

Lemma A.14 *Let P and P' be labeled processes such that $P \triangleright P'$. We have $\delta\ {}^{a}P = \delta\ {}^{a}P'$ and $\mathcal{E}(P) \equiv \mathcal{E}(P')$. Moreover, if P and P' are well-labeled, then $P \gg P'$ (and $P' \gg P$).*

Proof: The proof proceeds by induction on the structure of P. We observe that the cases of bang, prefix and nil are obvious since $P \triangleright P'$ implies $P = P'$. We show below the other cases.

- Suppose that $P = b[Q]$. By definition of $\triangleright$, we have $P' = b[Q']$ where $Q \triangleright Q'$. Hence, by induction hypothesis we have $\delta\ {}^{b}Q = \delta\ {}^{b}Q'$ and $\mathcal{E}(Q) \equiv \mathcal{E}(Q')$. Also, if Q and Q' are well-labeled, then $Q \gg Q'$. Using $\delta\ {}^{b}Q = \delta\ {}^{b}Q'$ we therefore obtain

  $$\delta\ {}^{a}P = (\{\ {}_{b}{}^{a}\}, \emptyset) \cup \delta\ {}^{b}Q = (\{\ {}_{b}{}^{a}\}, \emptyset) \cup \delta\ {}^{b}Q' = \delta\ {}^{a}P'.$$

  Moreover, $\mathcal{E}(Q) \equiv \mathcal{E}(Q')$ implies, by rule (Amb) of Table 3, $n[\mathcal{E}(Q)] \equiv n[\mathcal{E}(Q')]$ assuming $b = n_\lambda$. Suppose that P and P' are well-labeled. It means that Q and Q' also are well-labeled. Using $Q \gg Q'$ we derive $b[Q] \gg b[Q']$ by rule (Amb) of Table 10;

- Suppose that $P = Q_1 \mid Q_2$. The proof proceeds by induction similarly as in the preceding case.

- Suppose that $P = (\nu n_\lambda)\ Q$. By definition of $\triangleright$ there are two cases: either $P' = (\nu n_\lambda)\ Q'$ where $Q \triangleright Q'$ and $n \in fn(Q)$, or $n \notin fn(Q)$ and $P' = Q'$, where $Q \triangleright Q'$.

  1. Suppose that $P' = (\nu n_\lambda)\ Q'$ where $Q \triangleright Q'$. The proof proceeds by induction similarly as in the preceding case.

2. Suppose that $n \notin fn(Q)$ and $P' = Q'$, where $Q \triangleright Q'$. By induction hypothesis we have $\delta\ ^a Q = \delta\ ^a Q'$ and $\mathcal{E}(Q) \equiv \mathcal{E}(Q')$. Also, if Q and Q' are well-labeled we have $Q \gg Q'$. Using $n \notin fn(Q)$ we have immediately

$$\delta\ ^a P = \delta\ ^a (Q[m/n]) = \delta\ ^a Q = \delta\ ^a Q' = \delta\ ^a P'.$$

We observe also that $(\nu n)\mathcal{E}(Q) \equiv \mathcal{E}(Q)$ can be derived by applying the rules (Nil-Par), (Nil-Res) and (Res-Par) of Table 3 (using $n \notin fn(Q)$). Since $\mathcal{E}(Q) \equiv \mathcal{E}(Q')$, then we have also $\mathcal{E}(P) \equiv \mathcal{E}(Q')$. Similarly, for the case when when P and P' are well-labeled.

We conclude by observing that, when P and P' are well-labeled, $P \gg P'$ implies $P' \gg P$. In any case shown above only the symmetric rules of $\gg$ have been applied (see Table 10).

□

Soundness. The proof is rather complex; it is difficult in particular to reason about the well-labeling of the processes obtained in the inductive cases. We need some auxiliary properties. The following proposition shows a useful property of the reductions of well-labeled processes.

Proposition A.15 *Let P and Q be well-labeled processes such that $P \rightarrow Q$. If there exists λ such that $\lambda \in \Lambda(Q)$ and $H_{\mathcal{L}_I}(\lambda) \in fn(P)$, then there exists a well-labeled process Q', such that $P \rightarrow Q'$, $\lambda \notin \Lambda(Q')$ and $Q' \gg Q$.*

Proof: We first observe that by definition of well-labeling: $H_{\mathcal{L}_I}(\lambda) \in fn(P)$ implies $\lambda \notin \Lambda(P)$; analogously, $\lambda \in \Lambda(Q)$ implies $H_{\mathcal{L}_I}(\lambda) \notin n(Q)$. The proofs proceeds by induction on the depth of the inference of $P \rightarrow Q$. The cases of (In), (Out), and (Open) are immediate given that $\Lambda(Q) \subseteq \Lambda(P)$. In case (Cong) we have $P \gg P'$ and $P' \rightarrow P''$ and $P'' \gg Q$. If $\lambda \notin \Lambda(P'')$ we have finished. Otherwise, we observe that $H_{\mathcal{L}_I}(\lambda) \in fn(P')$, using $H_{\mathcal{L}_I}(\lambda) \in fn(P)$ and Proposition A.6. As P' is well-labeled, by induction hypothesis there exists R'' such that $P' \rightarrow R''$, $\lambda \notin \Lambda(R'')$ and $R'' \gg P'' \gg Q$. Hence, by rule (Cong) we derive $P \rightarrow R''$ such that $\lambda \notin \Lambda(R'')$, $R'' \gg Q$.
The other cases are similar and follow by induction hypothesis; we discuss as an example the case (Par). It means that $P = P_1 \mid P_2$ and $Q = P_1' \mid P_2$, where $P_1 \rightarrow P_1'$. We have $\lambda \in \Lambda(Q)$ and $\lambda \notin \Lambda(P)$, $H_{\mathcal{L}_I}(\lambda) \in fn(P)$ and $H_{\mathcal{L}_I}(\lambda) \notin fn(Q)$. Since P and Q are well-labeled, the only possibility is therefore that $\lambda \in \Lambda(P_1')$ and $H_{\mathcal{L}_I}(\lambda) \in fn(P_1)$. Hence, by induction hypothesis there exists P_1'' such that $P_1 \rightarrow P_1''$, $\lambda \notin \Lambda(P_1'')$ and $P_1'' \gg P_1'$. We observe that, by Proposition A.6, $P_1' \gg P_1''$ implies $fn(P_1'') = fn(P_1')$. Given $fn(P_1') = fn(P_1'')$ and $P_1' \mid P_2$ is well-labeled, there exists a re-indexing of labels ρ_I, such that $dom(\rho_I) = \Lambda(P_1'') \setminus \Lambda(P_1')$, $\lambda \notin \Lambda(P_1''\rho_I)$ and $Q' = P_1''\rho_I \mid P_2$ is well-labeled. By Propositions A.6 and A.7 we obtain both $P_1''\rho_I \gg P_1'$ and $P_1 \rightarrow P_1''\rho_I$. Since Q' is well-labeled, we derive $P_1 \mid P_2 \rightarrow Q'$ by applying rule (Par) to the premise $P_1 \rightarrow P_1''\rho_I$. Moreover, we have $P_1''\rho_I \mid P_2 \gg P_1' \mid P_2$ by applying rule (Par) to the premise $P_1''\rho_I \gg P_1'$.

□

To reason about the well-labeling of processes it is convenient to know precisely which new labels are introduced by a transition $S_1 \mapsto S_2$ between two states S_1 and S_2. To this aim we use $\texttt{new}(S_1 \mapsto S_2)$ to denote the set of labels which could have been introduced by an application of *new*, that is by the unfolding of replication. Formally,

1. $\mathtt{new}(S_1 \mapsto S_2) = \emptyset$ when $S_1 \mapsto S_2$ has been obtained by one of the rules **In**, **Out** or **Open**;
2. $\mathtt{new}(S_1 \mapsto S_2) = \Lambda(new_{S_1}(P))$ when $S_1 \mapsto S_2$ has been obtained by rule **Bang** and $S_2 = S_1 \cup \delta\ {}^{a}new_{S_1}(P)$.

The following proposition shows that the well-labeling of states is preserved by the transitions of Table 5 and clarifies in which sense the labels introduced by means of *new* in rule **Bang** are fresh.

Proposition A.16 *Let S_1 be a well-labeled state. If $S_1 \mapsto S_1'$, then S_1' is well-labeled. Moreover, assume that $S_1 \cup S_2$ is a well-labeled state such that $S_1 \cup S_2 \mapsto S_1' \cup S_2$. For any $\lambda \in \mathtt{new}(S_1 \mapsto S_1')$ we have $\lambda \notin \Lambda(S_2)$ and $H_{\mathcal{L}_I}(\lambda) \notin n(S_2)$.*

Proof: The proof is by cases on the rule applied to obtain $S_1 \mapsto S_1'$. Let $S_i = (T_i, C_i)$, for any $i \in \{1, 2\}$.

- As cases of **Open**, **In** and **Out** are similar, we discuss case **In** only. When $S_1 \mapsto S_1'$ has been obtained by rule **In**, we have $t = {}^{a}\mathtt{in}\, m_\gamma.P \in C_1$, ${}_{a}{}^{b}$, ${}_{m_\mu}{}^{b} \in T_1$ such that $a \neq m_\mu$. Moreover, $S_1' = S_1'' \cup \delta\ {}^{a}P$ where
$$S_1'' = ((T_1 \setminus \{\ {}_{a}{}^{b}\}) \cup \{\ {}_{a}{}^{m_\mu}\}, C_1 \setminus \{t\}).$$
Since S_1 is well-labeled and $t \in C_1$, then P is well-labeled and a is fresh for P. By Proposition A.13 we have that $\delta\ {}^{a}P$ is well-labeled. We also observe that S_1'' is well-labeled, as S_1 is well-labeled. Hence, S_1' is not well-labeled only when there exists a label λ, such that one of the following cases holds: (a) $\lambda \in \Lambda(\delta\ {}^{a}P) \setminus \Lambda(a)$ and either $\lambda \in \Lambda(S_1'')$ or $H_{\mathcal{L}_I}(\lambda) \in n(S_1'')$; (b) $\lambda \in \Lambda(S_1'')$ and $H_{\mathcal{L}_I}(\lambda) \in n(\delta\ {}^{a}P) \setminus n(a)$.
To discuss (a) and (b) we need to know the relation between the names and the labels of P and those of $\delta\ {}^{a}P$. By Proposition A.12, we have
    1. $\Lambda(\delta\ {}^{a}P) \setminus \Lambda(a) = \Lambda(P) \setminus \{\lambda \mid n_\lambda \in \mathcal{U}(P)\}$;
    2. $n(\delta\ {}^{a}P) \setminus n(a) = fn(P) \cup (bn(P) \setminus \{n \mid n_\lambda \in \mathcal{U}(P)\}) \cup \{H_{\mathcal{L}_I}(\lambda) \mid n_\lambda \in (\mathcal{U}(P) \setminus \mathcal{U}_u(P))\}$.

  We show case (b). Assume that $\lambda \in \Lambda(S_1'')$ and $H_{\mathcal{L}_I}(\lambda) \in n(\delta\ {}^{a}P) \setminus n(a)$. Given 2. we derive that either $H_{\mathcal{L}_I}(\lambda) \in n(P)$ or $H_{\mathcal{L}_I}(\lambda) \in \{H_{\mathcal{L}_I}(\lambda) \mid n_\lambda \in (\mathcal{U}(P) \setminus \mathcal{U}_u(P))\}$.
In the former case, since $H_{\mathcal{L}_I}(\lambda) \in n(P)$ and $t \in C_1$ we have $H_{\mathcal{L}_I}(\lambda) \in n(S_1)$. Moreover, we have $\Lambda(S_1'') \subseteq \Lambda(S_1)$. We obtain $\lambda \in \Lambda(S_1)$ and $H_{\mathcal{L}_I}(\lambda) \in n(S_1)$, which contradicts the well-labeling of S_1.
In the latter case, we have $\lambda \in \Lambda(P)$ and $\lambda \in \Lambda(S_1'')$. Hence, there is an object with label λ in S_1''. Since t has been removed from the configuration and $t \in C_1$, there are two objects with label λ in S_1, which contradicts again the well-labeling of S_1.
Case (a) follows by applying a similar argument using condition 1., $n(S_1'') \subseteq n(S_1)$ and $\Lambda(S_1'') \subseteq \Lambda(S_1)$.
Let $S_1 \cup S_2$ be a well-labeled state such that $S_1 \cup S_2 \mapsto S_1' \cup S_2$. We conclude by observing that $\mathtt{new}(S_1 \mapsto S_2) = \mathtt{new}(S_1 \cup S_2 \mapsto S_1' \cup S_2) = \emptyset$.
- Suppose that $S_1 \mapsto S_1'$ has been obtained by rule **Bang**. It means that $S_1' = (C_1, T_1) \cup \delta\ {}^{c}new_{S_1}(Q)$ for some ${}^{c}!Q \in C_1$. By definition, we have $new_{S_1}(Q) = Q\rho_I$ for a re-indexing of labels ρ_I such that $dom(\rho_I) = \Lambda(Q)$ and

1. $Q\rho_I$ is well-labeled;
2. there is no $\lambda \in \Lambda(Q\rho_I)$, such that either $\lambda \in \Lambda(S_1)$ or $H_{\mathcal{L}_I}(\lambda) \in n(S_1)$.

By conditions 1. and 2., $Q\rho_I$ is well-labeled and c is fresh for $Q\rho_I$. Consequently, by Proposition A.13, $\delta\ {}^c new_{S_1}(Q)$ is well-labeled. Since S_1 and $\delta\ {}^c new_{S_1}(Q)$ are well-labeled, S_1' is not well-labeled only when there exists λ such that one of the following cases holds: (a) $\lambda \in \Lambda(\delta\ {}^c new_{S_1}(Q)) \setminus \Lambda(c)$ and either $\lambda \in \Lambda(S_1)$ or $H_{\mathcal{L}_I}(\lambda) \in n(S_1)$; (b) $\lambda \in \Lambda(S_1)$ and $H_{\mathcal{L}_I}(\lambda) \in n(\delta\ {}^c new_{S_1}(Q)) \setminus n(c)$.

By Proposition A.12, we have

(i) $\Lambda(\delta\ {}^c new_{S_1}(Q)) \setminus \Lambda(c) = \Lambda(new_{S_1}(Q)) \setminus \{\lambda \mid n_\lambda \in \mathcal{U}(new_{S_1}(Q))\}$;

(ii) $n(\delta\ {}^c new_{S_1}(Q)) \setminus n(c) = fn(new_{S_1}(Q)) \cup (bn(new_{S_1}(Q)) \setminus \{n \mid n_\lambda \in \mathcal{U}(new_{S_1}(Q))\}) \cup \{H_{\mathcal{L}_I}(\lambda) \mid n_\lambda \in (\mathcal{U}(new_{S_1}(Q)) \setminus \mathcal{U}_u(new_{S_1}(Q)))\}$.

In case (a) we have $\lambda \in \Lambda(\delta\ {}^c new_{S_1}(Q)) \setminus \Lambda(c)$, and consequently $\lambda \in \Lambda(new_{S_1}(Q))$ using (i). When either $\lambda \in \Lambda(S_1)$ or $H_{\mathcal{L}_I}(\lambda) \in n(S_1)$ we have a contradiction with the requirement 2. above.

In case (b) we have $H_{\mathcal{L}_I}(\lambda) \in n(\delta\ {}^c new_{S_1}(Q)) \setminus n(c)$. Using (ii) we obtain that either $H_{\mathcal{L}_I}(\lambda) \in n(new_{S_1}(Q))$ or $\lambda \in \Lambda(new_{S_1}(Q))$. In the latter case, we have $\lambda \in \Lambda(new_{S_1}(Q))$ and $\lambda \in \Lambda(S_1)$, which contradicts the requirement 2. above. In the former case we have $H_{\mathcal{L}_I}(\lambda) \in n(new_{S_1}(Q))$ and $\lambda \in \Lambda(S_1)$. We observe that $n(Q) = n(new_{S_1}(Q))$ and ${}^c!Q \in C_1$. Hence, we have $H_{\mathcal{L}_I}(\lambda) \in n(S_1)$ and $\lambda \in \Lambda(S_1)$, which contradicts the well-labeling of S_1.

Let $S_1 \cup S_2$ be a well-labeled state such that $S_1 \cup S_2 \mapsto S_1' \cup S_2$. We observe that it is necessary to have $new_{S_1 \cup S_2}(Q) = Q\rho_I$, that is (besides condition 1. above): there is no $\lambda \in \Lambda(Q\rho_I)$, such that either $\lambda \in \Lambda(S_1 \cup S_2)$ or $H_{\mathcal{L}_I}(\lambda) \in n(S_1 \cup S_2)$. Given $\mathtt{new}(S_1 \mapsto S_1') = \Lambda(Q\rho_I)$, we have finished.

□

Now we show the main result of soundness.

Lemma A.17 *Let P be a well-labeled process and let $\delta\ {}^a P = S_1$ where $a \in \mathcal{A}$ is fresh for P. If $S_1 \mapsto S_2$, then there exists a well-labeled process Q, such that a is fresh for Q, $\delta\ {}^a Q = S_2$, $P \rightarrow_{\gg} Q$ and $\Lambda(Q) \setminus \Lambda(P) \subseteq \mathtt{new}(S_1 \mapsto S_2)$.*

Proof: The proof is by induction on the structure of P.

- Assume $P = 0$ or $P = M_\lambda.P_1$. We have $\delta\ {}^a 0 = (\emptyset, \emptyset) = S_1$ and $\delta\ {}^a M_\lambda.P_1 = (\emptyset, \{{}^a M_\lambda.P_1\}) = S_1$, respectively. In both cases the proof is trivial because there is no transition from S_1.
- Assume $P = !P_1$. We have $\delta\ {}^a !P_1 = (\emptyset, \{{}^a !P_1\}) = (T_1, C_1) = S_1$. Transition $S_1 \mapsto S_2$ could have been obtained only by applying rule **Bang**. It means that $S_2 = (\emptyset, \{{}^a !P_1\}) \cup \delta\ {}^a new_{S_1}(P_1)$. Let $Q = !P_1 \mid new_{S_1}(P_1)$. We observe that by definition of *new* and since $!P_1 \in C_1$, then Q is a well-labeled process. Therefore, by rule (Bang-Bang) of Table 10 we derive $!P_1 \gg !P_1 \mid new_{S_1}(P_1)$. We also have $\delta\ {}^a Q = (\emptyset, \{{}^a !P_1\}) \cup \delta\ {}^a new_{S_1}(P_1)$. We conclude by noticing that $\Lambda(Q) \setminus \Lambda(P) = \Lambda(new_{S_1}(P_1)) = \mathtt{new}(S_1 \mapsto S_2)$.

- Assume $P = (\nu n_\lambda)\, P_1$. We have $\delta\ ^a(\nu n_\lambda)\, P_1 = \delta\ \ ^aP'_1 = S_1$, where $m = H_{\mathcal{L}_I}(\lambda)$ and $P'_1 = P_1[m/n]$.

  Since P is well-labeled, $\lambda \notin \Lambda(P'_1)$, and consequently P'_1 is well-labeled. Hence, by induction hypothesis there exists a well-labeled process Q_1 such that $\delta\ ^aQ_1 = S_2$, $P'_1 \rightarrow_\gg Q_1$ and $\Lambda(Q_1) \setminus \Lambda(P'_1) \subseteq \mathtt{new}(S_1 \mapsto S_2)$.

  There are two cases: either $P'_1 \gg Q_1$ or $P'_1 \rightarrow Q_1$. We show only the latter one, the other being analogous. We show the existence of a well-labeled process Q, such that $P \rightarrow Q$, $\delta\ ^aQ = S_2$ and $\Lambda(Q) \setminus \Lambda(P) = \mathtt{new}(S_1 \mapsto S_2)$.

  The crucial observation to find out the right process Q is that Q_1 is a well-labeled process: it cannot be the case that $\lambda \in \Lambda(Q_1)$ and $m \in n(Q_1)$, where $m = H_{\mathcal{L}_I}(\lambda)$.

  1. Assume that $\lambda \notin \Lambda(Q_1)$. Let k be a new name, such that $k \neq m$ and $k \notin n(Q_1) \cup n(P_1)$ and there is no $\mu \in (\Lambda(Q_1) \cup \Lambda(P_1))$ with $H_{\mathcal{L}_I}(\mu) = k$. We take $Q = (\nu k_\lambda)\, Q_1[k/m]$. Since $\lambda \notin \Lambda(Q_1)$ we have also $\lambda \notin \Lambda(Q_1[k/m])$. Considering k has been properly chosen, Q is well-labeled. Moreover, we have

     $$\delta\ ^aQ = \delta\ ^a(Q_1[k/m][m/k]) = \delta\ ^aQ_1 = S_2.$$

     We now show that $P \rightarrow Q$. Since $P'_1 \rightarrow Q_1$ and k is a new name, we have also $P'_1[k/m] \rightarrow Q_1[k/m]$. Therefore, we derive $(\nu k_\lambda)P'_1[k/m] \rightarrow (\nu k_\lambda)Q_1[k/m]$ by applying rule (Res) to the premise $P'_1[k/m] \rightarrow Q_1[k/m]$. We also observe that $(\nu n_\lambda)\, P_1$ is α-convertible to $(\nu k_\lambda)\, P_1[m/n][k/m]$.

     It remains to show that $\Lambda(Q) \setminus \Lambda(P) \subseteq \mathtt{new}(S_1 \mapsto S_2)$. Since $\lambda \notin \Lambda(Q_1)$ we have $\Lambda(Q) \setminus \Lambda(P) = (\Lambda(Q_1[k/m]) \cup \{\lambda\}) \setminus (\Lambda(P_1) \cup \{\lambda\}) = \Lambda(Q_1) \setminus \Lambda(P'_1) \subseteq \mathtt{new}(S_1 \mapsto S_2)$.

  2. Assume that $\lambda \in \Lambda(Q_1)$ and $m \notin n(Q_1)$. We take $Q = Q_1$. Since Q_1 is well-labeled and $\delta\ ^aQ_1 = S_2$, it remains to show that $P \rightarrow Q_1$. The proof proceeds by considering the following two cases: $m \in fn(P'_1)$ or $m \notin fn(P'_1)$.

     When $m \notin fn(P'_1)$ we observe that $n \notin fn(P_1)$, that is $P'_1 = P_1$. Using $n \notin fn(P_1)$ we derive, by rules (Nil-Par), (Nil-Res) and (Res-Par), $(\nu n_\lambda)P_1 \gg P_1$. Since $P'_1 = P_1$ and $P'_1 \rightarrow Q_1$ we obtain by rule (Cong) $P \rightarrow Q_1$.

     If $m \in fn(P'_1)$ the proof is more complex. We use the fact that P'_1 is well-labeled, that is $\lambda \notin \Lambda(P'_1)$. Since $P'_1 \rightarrow Q_1$ and $m \notin fn(Q_1)$ we can apply Proposition A.15. We derive that there exists Q'_1 such that $Q'_1 \gg Q_1$, $P'_1 \rightarrow Q'_1$ and $\lambda \notin \Lambda(Q'_1)$.

     Since $\lambda \notin \Lambda(Q'_1)$, the process $(\nu k_\lambda)\, Q'_1[k/m]$ is well-labeled, where k is a new name chosen as in case 1. above. Moreover, by applying rule (Res) to the premise $P'_1 \rightarrow Q'_1$ we obtain $(\nu k_\lambda)\, P'_1[k/m] \rightarrow (\nu k_\lambda)\, Q'_1[k/m]$.

     We now deduce $(\nu k_\lambda)P'_1[k/m] \rightarrow Q_1$ from $(\nu k_\lambda)P'_1[k/m] \rightarrow (\nu k_\lambda)Q'_1[k/m]$. Since $Q'_1 \gg Q_1$ and $m \notin n(Q_1)$, then by Proposition A.6, $m \notin fn(Q'_1)$, that is $k \notin fn(Q'_1[k/m])$. Hence, by applying rules (Nil-Par), (Nil-Res) and (Res-Par) we obtain $(\nu k_\lambda)Q'_1[k/m] \gg Q'_1$. Using $Q'_1 \gg Q_1$ we have also $(\nu k_\lambda)\, Q'_1[k/m] \gg Q_1$. By rule (Cong) we therefore obtain $(\nu k_\lambda)\, P'_1[k/m] \rightarrow Q_1$. Moreover, we have that $(\nu n_\lambda)\, P_1$ is α-convertible to $(\nu k_\lambda)\, P_1[m/n][k/m]$.

     We conclude by observing that $\Lambda(Q) \setminus \Lambda(P) == \Lambda(Q_1) \setminus \Lambda(P) = \Lambda(Q_1) \setminus (\Lambda(P_1) \cup \{\lambda\})$. Since $\lambda \notin \Lambda(P_1)$ and $\Lambda(P_1) = \Lambda(P'_1)$ we have therefore $\Lambda(Q_1) \setminus \Lambda(P) \subseteq \Lambda(Q_1) \setminus \Lambda(P'_1) \subseteq \mathtt{new}(S_1 \mapsto S_2)$.

- Assume $P = b[P_1]$. We have $\delta\ {}^a b[P_1] = (\{\ {}_b^a\}, \emptyset) \cup \delta\ {}^b P_1 = S_1$. Transition $S_1 \mapsto S_2$ could have been obtained in two ways: either only P_1 contributes to the action or also ambient b participates. Notice that ambient a cannot be involved as a is fresh for P and P is well-labeled. This guarantees that S_1 is a well-labeled state with root a (see Proposition A.13). Let $S'_1 = \delta\ {}^b P_1 = (T'_1, C'_1)$.

    1. If only P_1 contributes to the action it means that $S'_1 \mapsto S'_2$ and $S_2 = S'_2 \cup (\{\ {}_b^a\}, \emptyset)$. As P is well-labeled, P_1 also is well-labeled and b is fresh for P_1. Therefore, by induction hypothesis there exist a well-labeled process Q_1, such that $\delta\ {}^b Q_1 = S'_2$, $P_1 \rightarrow_{\gg} Q_1$ and $\Lambda(Q_1) \setminus \Lambda(P_1) \subseteq \mathtt{new}(S'_1 \mapsto S'_2)$.

       There are two cases: either $P_1 \gg Q_1$ or $P_1 \rightarrow Q_1$. We show only the latter case, the other being analogous. The proof proceeds by showing that $b[Q_1]$ is well-labeled and that a is fresh for $b[Q_1]$. The well-labeling of Q is a necessary condition to derive a reduction $b[P_1] \rightarrow b[Q_1]$ by applying rule (Amb) to the premise $P_1 \rightarrow Q_1$. Let $Q = b[Q_1]$.

       Assume that either Q is not well-labeled or a is not fresh for Q. We recall that Q_1 is well-labeled and that $S_1 = S'_1 \cup (\{\ {}_b^a\}, \emptyset)$ is a well-labeled state. Therefore, the only possibility is that there exists a label λ, such that one of the following cases holds: (i) $\lambda \in \Lambda(Q_1)$ and either $\lambda \in \Lambda(a) \cup \Lambda(b)$ or $H_{\mathcal{L}_I}(\lambda) \in n(a) \cup n(b)$; (ii) $\lambda \in \Lambda(a) \cup \Lambda(b)$ and $H_{\mathcal{L}_I}(\lambda) \in n(Q_1)$.

       We consider before case (ii). Since the bound names of Q_1 can be α-converted, when needed, the interesting case is when $H_{\mathcal{L}_I}(\lambda) \in fn(Q_1)$. In this case we use $P_1 \rightarrow Q_1$ and we derive, by Proposition A.7, $fn(Q_1) \subseteq fn(P_1)$. Since $H_{\mathcal{L}_I}(\lambda) \in fn(Q_1)$ we have therefore $H_{\mathcal{L}_I}(\lambda) \in fn(P_1)$, and also $H_{\mathcal{L}_I}(\lambda) \in fn(P)$. Given that $\lambda \in \Lambda(a) \cup \Lambda(b)$ this contradicts either the well-labeling of P or the freshness of a for P.

       In case (i) we have $\lambda \in \Lambda(Q_1)$. We observe that it is not possible that $\lambda \in \Lambda(P)$. This because $\lambda \in \Lambda(a) \cup \Lambda(b)$ and $\lambda \in \Lambda(P)$ contradict either the well-labeling of P or the freshness of a for P. Similarly for $H_{\mathcal{L}_I}(\lambda) \in n(a) \cup n(b)$ and $\lambda \in \Lambda(P)$. Therefore, we have $\lambda \notin \Lambda(P_1)$ and $\lambda \in \Lambda(Q_1) \setminus \Lambda(P_1)$. We now use the fact that $\Lambda(Q_1) \setminus \Lambda(P_1) \subseteq \mathtt{new}(S'_1 \mapsto S'_2)$ and we deduce $\lambda \in \mathtt{new}(S'_1 \mapsto S'_2)$.

       We observe that $S_1 \mapsto S_2$, where $S_1 = S'_1 \cup (\{\ {}_b^a\}, \emptyset)$ ansd $S_2 = S'_2 \cup (\{\ {}_b^a\}, \emptyset)$. Therefore, by Proposition A.16, there is no $\mu \in \mathtt{new}(S'_1 \mapsto S'_2)$ such that either $\mu \in \Lambda((\{\ {}_b^a\}, \emptyset))$ or $H_{\mathcal{L}_I}(\mu) \in n((\{\ {}_b^a\}, \emptyset))$. Since $\Lambda((\{\ {}_b^a\}, \emptyset)) = \Lambda(a) \cup \Lambda(b)$ and $n((\{\ {}_b^a\}, \emptyset)) = n(a) \cup n(b)$ we have: $\lambda \in \mathtt{new}(S'_1 \mapsto S'_2)$ and either $\lambda \in \Lambda(a) \cup \Lambda(b)$ or $H_{\mathcal{L}_I}(\lambda) \in n(a) \cup n(b)$. This is a contradiction.

       Since Q is well-labeled, then a reduction $b[P_1] \rightarrow b[Q_1]$ can be obtained by applying rule (Amb) to the premise $P_1 \rightarrow Q_1$. Moreover, we have that a is fresh for Q and

       $$\delta\ {}^a Q = ((\{\ {}_b^a\}, \emptyset)) \cup \delta\ {}^b Q_1 = ((\{\ {}_b^a\}, \emptyset)) \cup S'_2 = S_2.$$

       It remains to show that $\Lambda(Q) \setminus \Lambda(P) \subseteq \mathtt{new}(S_1 \rightarrow S_2)$. This follows immediately using $\Lambda(Q_1) \setminus \Lambda(P_1) \subseteq \mathtt{new}(S'_1 \mapsto S'_2)$, $\mathtt{new}(S'_1 \mapsto S'_2) = \mathtt{new}(S_1 \mapsto S_2)$ and $\Lambda(Q_1) \setminus \Lambda(P_1) = \Lambda(Q) \setminus \Lambda(P)$ (as $\Lambda(b) \notin \Lambda(Q_1) \cup \Lambda(P_1)$).

    2. If both P_1 and b participate to the action, the only possibility is that some ambient c, which is top level inside b, goes out of b. It means that transition $S_1 \mapsto S_2$ has been obtained by rule **Out**. Therefore, there exist ${}_c^b \in T'_1$ and ${}^c\mathtt{out}\, n_\lambda . R \in C'_1$, such that

$b = n_\gamma$, and

$$S_1' = \delta\ {}^{b}T \cup (\{\ {}^{b}_{c}\}, \emptyset) \cup (\emptyset,\ {}^{c}\mathtt{out}\, n_\lambda.\, R) \cup \delta\ {}^{c}U$$

for some processes T and U. Moreover, the state S_2 reached from S_1 (by rule **Out**) is

$$S_2 = (\{\ {}^{a}_{b}\}, \emptyset) \cup (\{\ {}^{a}_{c}\}, \emptyset) \cup \delta\ {}^{b}T \cup \delta\ {}^{c}U \cup \delta\ {}^{c}R.$$

We now use $\delta\ {}^{b}P_1 = S_1'$ and the shape of S_1' to infer the structure of P_1. Examining the cases in the definition of δ, we observe that: the components $(\{\ {}^{b}_{c}\}, \emptyset)$ and $(\emptyset,\ {}^{c}\mathtt{out}\, n_\lambda.\, R)$ tell us that rules **DAmb** and **DPref** (possibly after rules **DRes** and **DPar**) have been used. Therefore, we have

$$P_1 \gg (\nu \tilde{p}_{\tilde{\mu}})\, (T' \mid c'[\mathtt{out}\, n_\lambda.\, R' \mid U'])$$

where $c = c'\eta$ and $T = T'\eta$, $U = U'\eta$ and $R = R'\eta$ for the substitution $\eta : \mathcal{N} \to \widehat{\mathcal{N}}_I$ such that $\eta(p) = H_{\mathcal{L}_I}(\mu)$.

Notice that we have grouped together the (eventual) unguarded restrictions by means of $\gg$. This result is based on the underlying assumption that the bound names $\tilde{p}$ can be α-converted and on the following properties due to the well-labeling if P: (i) $H_{\mathcal{L}_I}(\mu) \neq n$ for any $\mu \in \tilde{\mu}$; (ii) $n \notin \tilde{p}$. Condition (i) follows from $n \in n(P)$ using Proposition A.11. Condition (ii) follows from the fact that the restrictions are unguarded, since by Proposition A.12 any unguarded restriction is removed. Consequently, $n \in \tilde{p}$ implies $n \notin n(\delta\ {}^{b}P_1)$, which contradicts ${}^{c}\mathtt{out}\, n_\lambda.\, R \in C_1'$.

We now exploit the condition $n \notin \tilde{p}$ to derive, by applying rules (Amb) and (Res-Amb), that $P \gg P'$ where

$$P' = (\nu \tilde{p}_{\tilde{\mu}})\, (n_\gamma[T' \mid c'[\mathtt{out}\, n_\lambda.\, R' \mid U']]).$$

Let $Q = (\nu \tilde{p}_{\tilde{\mu}})\ b[T'] \mid c'[R' \mid U']$ which is obviously well-labeled. Moreover, we have $\delta^{a}Q = S_2$ and by rules (Out) and (Res) $P' \to Q'$. We therefore derive $P \to Q$ by applying rule (Cong).

We conclude by observing that $\Lambda(Q) \subseteq \Lambda(P)$. Thus, we have $\Lambda(Q) \setminus \Lambda(P) = \mathtt{new}(S_1 \mapsto S_2) = \emptyset$.

- Assume $P = P_1 \mid P_2$. We have $\delta\ {}^{a}P_1 \mid P_2 = \delta\ {}^{a}P_1\ \cup\ \delta\ {}^{a}P_2 = S_1$. Transition $S_1 \mapsto S_2$ could have been obtained in two ways: either only one of P_1 and P_2 participates to the action or the two processes interact with each other. In the latter case, we observe that ambient a cannot be involved as a is fresh for P. This guarantees that the topology is a tree with root a (see Proposition A.13). Therefore, $S_1 \mapsto S_2$ could have been obtained by the application either of rule **In** or of rule **Open**. In both cases the interaction may involve only processes and ambients which are top level inside a. Let $\delta\ {}^{a}P_1 = (T_1, C_1) = S_1'$ and $\delta\ {}^{a}P_2 = (T_2, C_2) = S_2'$.

  1. Suppose that only P_1 contributes to the action. We have $S_1 = S_1' \cup S_2'$ and $S_2 = S_1'' \cup S_2'$, where $S_1' \mapsto S_1''$. Since P is well-labeled and a is fresh for P, then also P_i is well-labeled and a is fresh for P_i, for any $i \in \{1, 2\}$. Hence, by induction hypothesis, we have $P_1 \to_{\gg} P_1'$ for a well-labeled process P_1', such that a is fresh for P_1', $\delta\ {}^{a}P_1' = S_1''$ and $\Lambda(P_1') \setminus \Lambda(P_1) \subseteq \mathtt{new}(S_1' \mapsto S_1'')$.

There are two cases: either $P_1 \gg P_1'$ or $P_1 \mapsto P_1'$. We show only the latter case, the other being analogous.

Similarly to the case of ambient we can apply rule (Par) to derive a transition $P_1 \mid P_2 \mapsto P_1' \mid P_2$ only when $P_1' \mid P_2$ is well-labeled. This case is however more complex as it may be the case that $P_1' \mid P_2$ is not well-labeled. We therefore consider a slightly different process $Q = P_1' \mid P_2'$, where $P_2 \triangleright P_2'$. We observe that, by definition of $\triangleright$, $fn(P_2) = fn(P_2')$ and $\Lambda(P_2') \subseteq \Lambda(P_2)$. Therefore, P_2' is well-labeled and a is fresh for P_2', as P_2 is well-labeled and a is fresh for P_2. Moreover, by Lemma A.14, we have $P_2 \gg P_2'$ and $\delta\ {}^a P_2' = \delta\ {}^a P_2 = S_2'$.

We now show that $Q = P_1' \mid P_2'$ is a well-labeled process. Assume that this is not the case. Since P_1' and P_2' are well-labeled the only possibility is that there exists a label λ such that one of the following cases hold: (i) $\lambda \in \Lambda(P_1')$ and either $\lambda \in \Lambda(P_2')$ or $H_{\mathcal{L}_I}(\lambda) \in n(P_2')$; (ii) $\lambda \in \Lambda(P_2')$ and $H_{\mathcal{L}_I}(\lambda) \in n(P_1')$.

We discuss before case (ii). Since the bound names of P_1' can be α-converted, when needed, the interesting case is when $H_{\mathcal{L}_I}(\lambda) \in fn(P_1')$. We use $P_1 \rightarrow P_1'$ and we obtain, by Proposition A.7, $fn(P_1') \subseteq fn(P_1)$. Hence, we have $H_{\mathcal{L}_I}(\lambda) \in fn(P_1)$. Given that $\Lambda(P_2'') \subseteq \Lambda(P_2)$ we obtain $\lambda \in \Lambda(P_2)$ and $H_{\mathcal{L}_I}(\lambda) \in fn(P_1)$. This contradicts the well-labeling of $P_1 \mid P_2$.

In case (i) we have $\lambda \in \Lambda(P_1')$. We observe that it cannot be the case that also $\lambda \in \Lambda(P_1)$. This because the well-labeling of $P_1 \mid P_2$ contradicts $\lambda \in \Lambda(P_1)$ and $\lambda \in \Lambda(P_2)$ (which follows from $\lambda \in \Lambda(P_2')$). Similarly for $\lambda \in \Lambda(P_1)$ and $H_{\mathcal{L}_I}(\lambda) \in n(P_2)$ (which follows from $H_{\mathcal{L}_I}(\lambda) \in n(P_2')$).

Therefore, we have $\lambda \notin \Lambda(P_1)$ and $\lambda \in \Lambda(P_1')$, that is $\lambda \in \Lambda(P_1') \setminus \Lambda(P_1)$. We now use the fact that $\Lambda(P_1') \setminus \Lambda(P_1) \subseteq \mathtt{new}(S_1' \mapsto S_1'')$ and we derive $\lambda \in \mathtt{new}(S_1' \mapsto S_1'')$.

We recall that $S_1 \mapsto S_2$, where $S_1 = S_1' \cup S_2'$ and $S_2 = S_1'' \cup S_2'$. Therefore, by Proposition A.16, there is no $\mu \in \mathtt{new}(S_1' \mapsto S_1'')$ such that either $\mu \in \Lambda(S_2')$ or $H_{\mathcal{L}_I}(\mu) \in n(S_2')$. Hence, it must be the case that (a) $\lambda \notin \Lambda(S_2')$ and (b) $H_{\mathcal{L}_I}(\lambda) \notin n(S_2')$.

We now use the fact that $\delta\ {}^a P_2' = S_2'$. By Proposition A.12, we have

– $\Lambda(S_2') \setminus \Lambda(a) = \Lambda(P_2') \setminus \{\lambda \mid n_\lambda \in \mathcal{U}(P_2')\}$;

– $n(S_2') \setminus n(a) = fn(P_2') \cup (bn(P_2') \setminus \{n \mid n_\lambda \in \mathcal{U}(P_2')\}) \cup \{H_{\mathcal{L}_I}(\lambda) \mid n_\lambda \in (\mathcal{U}(P_2') \setminus \mathcal{U}_u(P_2'))\}$.

Using the results above, we now show that both possibilities $\lambda \in \Lambda(P_2')$ and $H_{\mathcal{L}_I}(\lambda) \in n(P_2')$ contradicts either (a) or (b).

Assume that $H_{\mathcal{L}_I}(\lambda) \in n(P_2')$. As usual the interesting case is when $H_{\mathcal{L}_I}(\lambda) \in fn(P_2')$. Given the previous conditions we have $fn(P_2') \subseteq n(S_2')$. Therefore, $H_{\mathcal{L}_I}(\lambda) \in fn(P_2')$ implies $H_{\mathcal{L}_I}(\lambda) \in n(S_2')$ which contradicts (b).

Assume that $\lambda \in \Lambda(P_2')$. Given the previous conditions we have two possibilities: either $\lambda \in \Lambda(S_2')$ or $n_\lambda \in \mathcal{U}(P)$. The former case contradicts immediately (a). In the latter case, we use $P_2 \triangleright P_2'$, which says that P_2' has no unguarded and unnecessary restrictions ($\mathcal{U}_u(P_2') = \emptyset$). Consequently, when $n_\lambda \in \mathcal{U}(P)$, then $H_{\mathcal{L}_I}(\lambda) \in n(S_2')$. This contradicts condition (b).

We now show that there exists a reduction $P_1 \mid P_2 \rightarrow Q$, where $Q = P_1' \mid P_2'$. We observe that $\Lambda(P_2') \subseteq \Lambda(P_2)$, and thus $P_1 \mid P_2'$ is well-labeled since $P_1 \mid P_2$ is well-labeled. Since also $P_1' \mid P_2'$ is well-labeled, by applying rule (Par) to the premise $P_1 \rightarrow P_1'$, we obtain $P_1 \mid P_2' \rightarrow P_1' \mid P_2'$. Since $P_2 \gg P_2'$ we have also $P_1 \mid P_2 \gg P_1 \mid P_2'$. We therefore derive $P_1 \mid P_2 \rightarrow P_1' \mid P_2'$ by applying rule (Cong).

Moreover, it is immediate to check that

$$\delta\ {}^{a}P_1' \mid P_2' = S_1'' \cup S_2' = S_2.$$

It remains to show that $\Lambda(Q) \setminus \Lambda(P) \subseteq \mathtt{new}(S_1 \to S_2)$. We observe that, since $P_1 \mid P_2$ and $P_1' \mid P_2'$ are well-labeled, $\Lambda(P_1) \cap \Lambda(P_2) = \emptyset$ and $\Lambda(P_1') \cap \Lambda(P_2') = \emptyset$. Moreover, $\Lambda(P_2') \subseteq \Lambda(P_2)$. Therefore, $\Lambda(Q) \setminus \Lambda(P) = (\Lambda(P_1') \cup \Lambda(P_2')) \setminus (\Lambda(P_1) \cup \Lambda(P_2)) = \Lambda(P_1') \setminus (\Lambda(P_1) \cup \Lambda(P_2)) \subseteq \Lambda(P_1') \setminus \Lambda(P_1)$. We conclude because $\Lambda(P_1') \setminus \Lambda(P_1) \subseteq \mathtt{new}(S_1' \mapsto S_1'')$ and $\mathtt{new}(S_1' \mapsto S_1'') = \mathtt{new}(S_1 \mapsto S_2)$.

2. Suppose that rule **In** has been applied. We have either ${}_{b}{}^{a} \in T_1$, ${}^{b}\mathtt{in}\,m_\lambda.\,R \in C_1$ and ${}_{m_\mu}{}^{a} \in T_2$ or the converse. Suppose the former case holds. It means that S_1' and S_2' have the following shape, respectively

$$S_2' = \delta^{a}W \cup (\ {}_{m_\mu}{}^{a}, \emptyset) \cup \delta^{m_\mu}V$$

$$S_1' = \delta^{a}U \cup (\ {}_{b}{}^{a}, \emptyset) \cup (\emptyset,\ {}^{b}\mathtt{in}\,m_\lambda.\,R) \cup \delta^{b}T$$

for some processes W,V,U, and T. Moreover, the state S_2 reached from S_1 (by rule **In**) is

$$S_2 = (\ {}_{b}{}^{m_\mu}, \emptyset) \cup (\ {}_{m_\mu}{}^{a}, \emptyset) \cup \delta^{a}U \cup \delta^{b}R \cup \delta^{b}T \cup \delta^{a}W \cup \delta^{m_\mu}V.$$

Since $\delta\ {}^{a}P_1 = S_1'$ and $\delta\ {}^{a}P_2 = S_2'$, we argue that (reasoning on the definition of δ, similarly to case **Out**)

$$P_1 \gg (\nu\tilde{p}_{\tilde{\mu}})\,(U' \mid b'[\mathtt{in}\,m_\lambda.\,R' \mid T'])$$

$$P_2 \gg (\nu\tilde{q}_{\tilde{\nu}})\,(W' \mid m_\mu[V'])$$

where $b = b'\eta_1$, $T = T'\eta_1$, $U = U'\eta_1$ and $R = R'\eta_1$ and $W = W'\eta_2$ and $V = V'\eta_2$ for the substitutions $\eta_i : \mathcal{N} \to \widehat{\mathcal{N}}_I$ where $\eta_1(p) = H_{\mathcal{L}_I}(\mu)$ and $\eta_2(q) = H_{\mathcal{L}_I}(\nu)$.
We now notice that it cannot be the case that $m \in \tilde{p}$ or $m \in \tilde{q}$. Suppose that $m \in \tilde{p}$. Since the restrictions $(\nu\tilde{p}_{\tilde{\mu}})$ are unguarded and P_1 is well-labeled, then by Proposition A.12, we obtain $m \notin n(\delta\ {}^{a}P_1)$, which contradicts ${}^{b}\mathtt{in}\,m_\lambda.\,R \in C_1$. Similarly, using the well-labeling of P_2, $m \in \tilde{q}$ contradicts ${}_{m_\mu}{}^{a} \in T_2$.
Therefore, we may assume without loss of generality that $\tilde{p} \cap \tilde{q} = \emptyset$, and we have $P_1 \mid P_2 \gg P'$ where

$$P' = (\nu\tilde{p}_{\tilde{\mu}}, \tilde{q}_{\tilde{\nu}})\,(U' \mid b'[\mathtt{in}\,m_\lambda.\,R' \mid T'] \mid m_\gamma[V'] \mid W').$$

Let

$$Q = (\nu\tilde{p}_{\tilde{\mu}}, \tilde{q}_{\tilde{\nu}})\,(U' \mid W' \mid m_\mu[V' \mid b[R' \mid T']]).$$

It is obvious that Q is a well-labeled process and $\delta\ {}^{a}Q = S_2$. Also, we have by rules (In), (Par) and (Res) $P' \to Q$. We therefore derive $P \to Q$ by applying rule (Cong).
We conclude by observing that $\Lambda(Q) \subseteq \Lambda(P)$. Thus, we have $\Lambda(Q) \setminus \Lambda(P) = \mathtt{new}(S_1 \mapsto S_2) = \emptyset$.

3. The case when rule **Open** has been applied is similar to that of rule **In** above.

□

Completeness. To show completeness we need some auxiliary properties. The following lemma shows the relation between the states representing two well-labeled processes which are structural congruent.

Lemma A.18 *Let P and Q be well-labeled processes and let $a \in \mathcal{A}$, such that a is fresh for P and Q. If $P \gg Q$, then either $\delta\ {}^{a}P = \delta\ {}^{a}Q$ or $\delta\ {}^{a}P \mapsto \delta\ {}^{a}Q$.*

Proof: By induction on the depth of $P \gg Q$. It is easy to check that in any case of Table 10 the states obtained via δ are equal apart from the case (Bang-Bang). In case (Bang-Bang) we have $P = !R$ and $Q = !R \mid new(R)$. Hence, we have $\delta\ {}^{a}P = S_1 = (\emptyset,\ {}^{a}!R)$ and $\delta\ {}^{a}Q = S_2 = (\emptyset,\ {}^{a}!R) \cup \delta\ {}^{a}new(R)$. We observe that $\delta\ {}^{a}P \mapsto S_2$ by rule **Bang**.

□

Proposition A.19 *Let S_1 be a well-labeled state such that $S_1 \mapsto S_1'$. If S_2 is a well-labeled state such that $S_1 \cup S_2$ and $S_1' \cup S_2$ is well-labeled, then we have also $S_1 \cup S_2 \mapsto S_1' \cup S_2$.*

Proof: The proof is by cases on the rule applied to derive $S_1 \mapsto S_1'$. The cases of **In**, **Out** and **Open** are trivial; the side conditions impose constraints which hold also for $S_1 \cup S_2$. In the case **Bang** instead we have $S_1 = (T_1, C_1)$ and $S_1' = (T_1, C_1) \cup \delta\ {}^{c}new_{S_1}(Q)$ for some ${}^{c}Q \in C_1$. We obtain $S_1 \cup S_2 \mapsto S_1' \cup S_2$, as $new_{S_1}(Q) = new_{S_1 \cup S_2}(Q)$ is ensured by the well-labeling of $S_1' \cup S_2$.

□

Lemma A.20 *Let P be a well-labeled process such that $P \to Q$. For any $c \in \mathcal{A}$ which is fresh for P, we have $\delta\ {}^{c}P \mapsto^* \delta\ {}^{c}Q$.*

Proof: The proof is by induction on the depth of the derivation of $P \to Q$. The last rule used could have been (In), (Out), (Open), one of the structural rules (Res),(Par),(Amb) or rule (Cong).

- Assume that $P \to Q$ has been obtained by applying rule (In). It means that $P = a[\mathtt{in}\, m_\lambda. P' \mid Q'] \mid b[R']$, where $a = n_\mu$ and $b = m_\gamma$, and $Q = b[a[P' \mid Q'] \mid R']$.
  By definition of δ we have
  $$\delta\ {}^{c}P = \delta\ {}^{c}a[\mathtt{in}\, m_\lambda. P' \mid Q'] \cup \delta\ {}^{c}b[R'] =$$
  $$(\{\ {}_{a}^{c},\ {}_{b}^{c}\}, \emptyset) \cup (\emptyset,\ {}^{a}\mathtt{in}\, m_\lambda. P') \cup \delta\ {}^{a}Q' \cup \delta\ {}^{b}R'.$$
  Therefore, by applying rule **In** we obtain a transition $\delta\ {}^{c}P \mapsto S$ where
  $$S = (\{\ {}_{a}^{b},\ {}_{b}^{c}\}, \emptyset) \cup \delta\ {}^{a}Q' \cup \delta\ {}^{b}R' \cup \delta\ {}^{a}P'.$$
  We conclude by observing that, by definition of δ,
  $$\delta\ {}^{c}Q = (\{\ {}_{b}^{c}\}, \emptyset) \cup \delta\ {}^{b}(a[P' \mid Q'] \mid R') = (\{\ {}_{a}^{b},\ {}_{b}^{c}\}, \emptyset) \cup \delta\ {}^{a}P' \cup \delta\ {}^{a}Q' \cup \delta\ {}^{b}R' = S.$$

- Assume that $P \to Q$ has been obtained by applying rule (Out). It means that $P = b[a[\mathtt{out}\, m_\lambda.\, P' \mid Q'] \mid R']$, where $a = n_\mu$ and $b = m_\gamma$, and $Q = b[R'] \mid a[P' \mid Q']$.
  By definition of δ we have
  $$\delta\ {}^cP = (\{\ {}_b^c\}, \emptyset) \cup \delta\ {}^b(a[\mathtt{out}\, m_\lambda.\, P' \mid Q'] \mid R') =$$
  $$(\{\ {}_b^c,\ {}_a^b\}, \emptyset) \cup (\emptyset,\ {}^a\mathtt{out}\, m_\lambda.\, P') \cup \delta\ {}^aQ' \cup \delta\ {}^bR'.$$
  Moreover, by applying rule **Out** we obtain a transition $\delta\ {}^cP \mapsto S$ where
  $$S = (\{\ {}_b^c,\ {}_a^c\}, \emptyset) \cup \delta\ {}^aP' \cup \delta\ {}^aQ' \cup \delta\ {}^bR'.$$
  We conclude by observing that, by definition of δ,
  $$\delta\ {}^cQ = \delta\ {}^cb[R'] \cup \delta\ {}^ca[P' \mid Q'] = (\{\ {}_b^c,\ {}_a^c\}) \cup \delta\ {}^bR' \cup \delta\ {}^aP' \mid Q' = S.$$
- Assume $P \to Q$ has been obtained by applying rule (Open). It means that $P = \mathtt{open}\, n_\lambda.\, P' \mid a[R']$, where $a = n_\mu$, and $Q = P' \mid R'$.
  By definition of δ we have
  $$\delta\ {}^cP = \delta\ {}^c\mathtt{open}\, n_\lambda.\, P' \cup \delta\ {}^ca[R'] = (\ {}_a^c, \emptyset) \cup (\emptyset,\ {}^c\mathtt{open}\, n_\lambda.\, P') \cup \delta^aR'.$$
  Moreover, by applying rule **Open** we obtain a transition $\delta\ {}^cP \mapsto S$ where
  $$S = \delta\ {}^cP' \cup (T[\ {}_d^c/\ {}_d^a], C[\ {}^cR/\ {}^aR])$$
  $$\delta^aR' = (T, C)$$
  We conclude by observing that, by definition of δ
  $$\delta\ {}^cQ = \delta\ {}^cP' \cup \delta\ {}^cR'.$$
  Since c is fresh, we also have $(T[\ {}_d^c/\ {}_d^a], C[\ {}^cR/\ {}^aR]) = \delta\ {}^cR'$. We therefore conclude $\delta\ {}^cQ = S$.
- Assume $P \to Q$ has been obtained by applying rule (Amb). It means that $P = a[P_1]$, where $a = n_\lambda$, and $Q = a[P_2]$, where $P_1 \to P_2$. By definition of δ we have
  $$\delta\ {}^cP = \delta\ {}^ca[P_1] = (a^c, \emptyset) \cup \delta\ {}^aP_1.$$
  Since P is well-labeled, then P_1 is well-labeled and a is fresh for P_1. Hence, by induction hypothesis we have $\delta\ {}^aP_1 \mapsto^* S'$, where $\delta\ {}^aP_2 = S'$. We now observe that $Q = a[P_2]$ is well-labeled. Hence, by Proposition A.13, we have that $\delta\ {}^cQ$ is well-labeled. Also, by definition of δ we have
  $$\delta\ {}^cQ = (a^c, \emptyset) \cup \delta\ {}^aP_2 = (a^c, \emptyset) \cup S'.$$
  We conclude by applying Proposition A.19. Since $(a^c, \emptyset) \cup S'$ is well-labeled and $\delta\ {}^aP_1 \mapsto^* S'$, then we have also
  $$\delta\ {}^cP \mapsto^* (a^c, \emptyset) \cup S'.$$

- Assume $P \to Q$ has been obtained by applying rule (Par). It means that $P = P_1 \mid P_2$ and $Q = P_1' \mid P_2$, where $P_1 \to P_1'$. By definition of δ we have

  $$\delta\ {}^cP = \delta\ {}^cP_1 \cup \delta\ {}^cP_2.$$

  Since P is well-labeled and c is fresh for P, then also P_1 is well-labeled and c is fresh for P_1. Hence, by induction hypothesis we have $\delta\ {}^cP_1 \mapsto^* S'$, where $\delta\ {}^cP_1' = S'$.

  We now observe that $Q = P_1' \mid P_2$ is well-labeled. Hence, by Proposition A.13, we have that $\delta\ {}^cQ$ is well-labeled. Also, by definition of δ we have

  $$\delta\ {}^cQ = \delta\ {}^cP_1' \cup \delta\ {}^cP_2 = S' \cup \delta\ {}^cP_2.$$

  We conclude by applying Proposition A.19. Since $S' \cup \delta\ {}^cP_2$ is well-labeled and $\delta\ {}^cP_1 \mapsto^* S'$, then we have also

  $$\delta\ {}^cP \mapsto^* S' \cup \delta\ {}^cP_2.$$

- Assume $P \to Q$ has been obtained by applying rule (Res). It means that $P = (\nu n_\lambda)\, P_1$ and $Q = (\nu n_\lambda)\, P_2$ where $P_1 \to P_2$. By definition of δ we have

  $$\delta\ {}^cP = \delta\ {}^c(P_1[m/n])$$

  where $m = H_{\mathcal{L}_I}(\lambda)$.

  We observe that since P is well-labeled, then $m \notin n(P_1)$. Since $P_1 \to P_2$, then by Proposition A.7, $fn(P_2) \subseteq fn(P_1)$, and consequently also $m \notin n(P_2)$. Considering the bound names can be α-converted, if needed, we derive $P_1[m/n] \to P_2[m/n]$ from $P_1 \to P_2$.

  Since P is well-labeled, then $\lambda \notin \Lambda(P_1[m/n])$. Consequently, $P_1[m/n]$ is a well-labeled process. Therefore, by induction hypothesis we have $\delta\ {}^c(P_1[m/n]) \mapsto^* S'$, where $\delta\ {}^c(P_2[m/n]) = S'$. We conclude by observing that

  $$\delta\ {}^cQ = \delta\ {}^c(P_2[m/n]) = S'.$$

- Assume $P \to Q$ has been obtained by applying rule (Cong). It means that $P_1 \to Q_1$ for some processes P_1, Q_1, such that $P \gg P_1$ and $Q_1 \gg Q$ By induction hypothesis we have $\delta\ {}^cP_1 \mapsto^* S$ where $S = \delta\ {}^cQ_1$. By Lemma A.18, we have $\delta\ {}^cP \mapsto^* S$. Again by Lemma A.18, we have either $\delta\ {}^cQ_1 = \delta\ {}^cQ$ or $\delta\ {}^cQ_1 \mapsto \delta\ {}^cQ$. In both cases $\delta\ {}^cP \mapsto^* \delta\ {}^cQ$.

□

A.3 Equivalence

We show the proof of Theorem 4.5.

Soundness: if $\delta\ {}^aP \mapsto S$, then by Lemma A.17 there exists a well-labeled process Q, such that $\delta\ {}^aQ = S$ and $P \to_\gg Q$. By Lemmas A.1 and A.2 we have $\mathcal{E}(P) \to_\equiv \mathcal{E}(Q)$.

Completeness: if $\mathcal{E}(P) \to Q$, then by Lemma A.9 there exists a well-labeled process Q', such that $\mathcal{E}(Q') \equiv Q$ and $P \to Q'$. By Lemma A.20 we have $\delta\ {}^aP \mapsto^* \delta\ {}^aQ'$.

B Safeness of the abstractions

The following proposition recalls some well-known results of domain theory which are useful in the proofs.

Proposition B.1

1. *Given any set $\mathcal{S}$, $\langle \wp(\mathcal{S}), \subseteq \rangle$ is a complete lattice.*
2. *Given two complete lattices $\langle \mathcal{S}_1, \subseteq_1 \rangle$, $\langle \mathcal{S}_2, \subseteq_2 \rangle$, the product $\langle (\mathcal{S}_1 \times \mathcal{S}_2), \subseteq_{cw} \rangle$, where $\subseteq_{cw}$ is the component-wise induced ordering, is a complete lattice.*

B.1 First Abstraction

We first show that the pair of functions $(\alpha^\diamond, \gamma^\diamond)$ forms a Galois connection between $\langle \mathcal{S}^\natural, \subseteq \rangle$ and $\langle \mathcal{S}^\diamond, \subseteq^\diamond \rangle$ (Theorem 5.6).

Proposition B.2 *The concrete domain $\langle \mathcal{S}^\natural, \subseteq \rangle$ and the abstract domain $\langle \mathcal{S}^\diamond, \subseteq^\diamond \rangle$ are* complete lattices.

Proof: The concrete domain $\mathcal{S}^\natural = \wp(\mathcal{S}_{/\sim})$ is a complete lattice by case 1. of Proposition B.1. The abstract domain $\langle \mathcal{S}^\diamond, \subseteq^\diamond \rangle$ is a complete lattice by case 2. of Proposition B.1. Notice that, by definition of $\subseteq^\diamond$ (Definition 5.3), given two well-labeled states $S_1^\diamond$ and $S_2^\diamond$, $S_1^\diamond \cup S_2^\diamond$ is a well-labeled state as well.

□

The following proposition states the basic properties of the concretization and abstraction functions.

Proposition B.3 *Function $\alpha^\diamond : \langle \mathcal{S}^\natural, \subseteq \rangle \to \langle \mathcal{S}^\diamond, \subseteq^\diamond \rangle$ is* monotonic *and* continuous *and function $\gamma^\diamond : \langle \mathcal{S}^\diamond, \subseteq^\diamond \rangle \to \langle \mathcal{S}^\natural, \subseteq \rangle$ is* monotonic.

Proof: Straightforward by Definition 5.5. □

The properties stated above are enough to prove Theorem 5.6.

Proof: [of Theorem 5.6] We show that $(\alpha^\diamond, \gamma^\diamond)$ is a Galois connection (see Definition 2.1). By Proposition B.2 the concrete and abstract domains are complete lattices. Also, by Proposition B.3 both $\alpha^\diamond$ and $\gamma^\diamond$ are monotonic. Hence, it remains two show that, for $S^\diamond \in \mathcal{S}^\diamond$ and $S^\natural \in \mathcal{S}^\natural$, we have

$$S^\natural \subseteq \gamma^\diamond(\alpha^\diamond(S^\natural))$$

$$\alpha^\diamond(\gamma^\diamond(S^\diamond)) \subseteq^\diamond S^\diamond$$

Both assertions follow rather obviously from Definition 5.5. We have $S^\natural \subseteq \gamma^\diamond(\alpha^\diamond(S^\natural))$ since by definition of $\gamma^\diamond$ and $\alpha^\diamond$,

$$\gamma^\diamond(\alpha^\diamond(S^\natural)) = \bigcup \{[S] \mid \alpha^\diamond(\{[S]\}) \subseteq^\diamond \textstyle\bigcup^\diamond_{[S] \in S^\natural} \alpha^\diamond([S])\}.$$

Moreover, by definition of $\alpha^\diamond$ and $\gamma^\diamond$, and by continuity of $\alpha^\diamond$ (Proposition B.3) we have

$$\alpha^\diamond(\gamma^\diamond(S^\diamond)) = \alpha^\diamond(\bigcup\{[S] \mid \alpha^\diamond(\{[S]\})\subseteq^\diamond S^\diamond\}) = \bigcup^\diamond \alpha^\diamond(\{[S] \mid \alpha^\diamond(\{[S]\})\subseteq^\diamond S^\diamond\}).$$

By definition of least upper bound on a complete lattice we conclude therefore

$$\bigcup^\diamond \alpha^\diamond(\{[S] \mid \alpha^\diamond(\{[S]\})\subseteq^\diamond S^\diamond\})\subseteq^\diamond S^\diamond.$$

□

We now show some basic properties of the concrete and abstract semantic functions which are needed to establish the safeness of the abstraction (Lemma 5.8).

Lemma B.4 *Let $S_1^\diamond, S_2^\diamond \in \mathcal{S}^\diamond$ be well-labeled abstract states such that $S_1^\diamond \subseteq^\diamond S_2^\diamond$. if $S_1^\diamond \mapsto^\diamond S'^\diamond_1$, then there exists a transition $S_2^\diamond \mapsto^\diamond S'^\diamond_2$, such that $S'^\diamond_1 \subseteq^\diamond S'^\diamond_2$.*

Proof: There are two cases depending on whether $S_1^\diamond \subseteq S_2^\diamond$ or not. In the former case the proof is straightforward. In the latter case, it means that there exists an abstract state $S''^\diamond_1$, such that $S''^\diamond_1 = S_1^\diamond \rho$ for a renaming $\rho : \mathcal{L}_I \to \mathcal{L}^\diamond$, where either $\rho(\ell_1) = \ell_1$ or $\rho(\ell_1) = \ell_\omega$, and $S''^\diamond_1 \subseteq S_2^\diamond$. It is easy to check (by cases on the rules of Table 7) that we have $S''^\diamond_1 \mapsto^\diamond S'''^\diamond_1$ such that $S'^\diamond_1 \subseteq^\diamond S'''^\diamond_1$. Since $S''^\diamond_1 \subseteq S_2^\diamond$ and $S''^\diamond_1 \mapsto^\diamond S'''^\diamond_1$, then we have also $S_2^\diamond \mapsto^\diamond S'^\diamond_2$ such that $S'''^\diamond_1 \subseteq^\diamond S'^\diamond_2$. We conclude because $S'^\diamond_1 \subseteq^\diamond S'''^\diamond_1 \subseteq^\diamond S'^\diamond_2$.

□

Lemma B.5 *Let $S \in \mathcal{S}$ and $S^\diamond \in \mathcal{S}^\diamond$. The functions $\Psi_S : \langle \mathcal{S}^\natural, \subseteq\rangle \to \langle \mathcal{S}^\natural, \subseteq\rangle$ and $\Psi^\diamond_{S^\diamond} : \langle \mathcal{S}^\diamond, \subseteq^\diamond\rangle \to \langle \mathcal{S}^\diamond, \subseteq^\diamond\rangle$ are monotonic.*

Proof: The proof follows immediately by Lemma B.4 using Definitions 4.9 and 5.7.

□

We state some relevant properties of the auxiliary abstraction function $\alpha^\diamond : \mathcal{S} \to \mathcal{S}^\diamond$ which maps a state into an abstract state (see Definition 5.5, case 1.). The following lemma says that $\alpha^\diamond$ is continuous for union of states with a special shape (Recall that the abstraction over sets of states $\alpha^\diamond : \mathcal{S}^\natural \to \mathcal{S}^\diamond$ is continuous as shown by Proposition B.3). To state formally this result we need to introduce an auxiliary concept. Let S_1, S_2 be two well-labeled states. We say that S_2 is a *sub-tree* of S_1 with root $a \in \mathcal{A}$ iff a is the root of S_2, and only ambient a occurs both in S_1 and S_2.

We introduce a convention which is useful in the following proofs. We recall that any object may have several abstractions depending on the global number of occurrences of its labels in the state. In the abstraction $\alpha^\diamond$ (see Definition 5.5) this is formalised by: the renaming $\rho^\diamond_S$, which depend on the state S and introduce the multiplicity counting the indexes; the substitution $\eta^\diamond$ which simply removes indexes. When the renaming $\rho^\diamond_S$ is clear from the context we may use: $a^\diamond$ to denote the abstract version of a; $P^\diamond$ to denote that abstract version of P.

Lemma B.6 *Let S_1, S_2 be two well-labeled states, such that $S_1 \cup S_2$ also is well-labeled. If S_2 is a sub-tree of S_1 with root a, then we have*

$$\alpha^\diamond(S_1 \cup S_2) = \alpha^\diamond(S_1) \cup^\diamond \alpha^\diamond(S_2)\{\![a^{\diamond b^\diamond}/a^{\diamond @}]\!\}$$

where b is the father of a in S_1 [16].

[16]Meaning that $a^b \in T_1$ for $S_1 = (T_1, C_1)$.

Proof: Let $\alpha^\diamond(S_1 \cup S_2) = (T^\diamond, C^\diamond)$, $S_i = (T_i, C_i)$ and $\alpha^\diamond(S_i) = ({T_i}^\diamond, {C_i}^\diamond)$ for $i \in \{1,2\}$. We recall that, by definition of $\alpha^\diamond$ (Definition 5.5), we have $(T^\diamond, C^\diamond) = (T'^\diamond, C'^\diamond)\rho^\diamond_{S_1 \cup S_2}\eta^\diamond$ where

$$T'^\diamond = \{\ {}_{a}^{b^{c}} \mid\ {}_{a}^{b},\ {}_{b}^{c} \in T_1 \cup T_2\}$$

$$C'^\diamond = \{\ {}^{{}_{a}^{b}}P \mid\ \ {}_{a}^{b} \in T_1 \cup T_2,\ {}^{a}P \in C_1 \cup C_2\}.$$

Analogously, for $i \in \{1,2\}$, we have $(T_i^\diamond, C_i^\diamond) = ({T'_i}^\diamond, {C'_i}^\diamond)\rho^\diamond_{S_i}\eta^\diamond$ where

$${T'_i}^\diamond = \{\ {}_{a}^{b^{c}} \mid\ {}_{a}^{b},\ {}_{b}^{c} \in T_i\}$$

$${C'_i}^\diamond = \{\ {}^{{}_{a}^{b}}P \mid\ \ {}_{a}^{b} \in T_i,\ {}^{a}P \in C_i\}.$$

We first show that $T'^\diamond \subseteq {T'_1}^\diamond \cup {T'_2}^\diamond\{\![a^b/a^@]\!\}$. Let us consider a generic element $c^{d^e} \in T'^\diamond$. It means that ${}_{c}^{d},\ {}_{d}^{e} \in T_1 \cup T_2$. There are several possibilities:

1. Both ${}_{c}^{d} \in T_1$ and ${}_{d}^{e} \in T_1$. It is immediate to check that we have also $c^{d^e} \in {T'_1}^\diamond$.
2. Both ${}_{c}^{d} \in T_2$ and ${}_{d}^{e} \in T_2$. Similarly as in the previous case we have $c^{d^e} \in {T'_2}^\diamond$. We now observe that c and d cannot be a, because a is the root of S_2. We therefore conclude that $c^{d^e} \in {T'_2}^\diamond\{\![a^b/a^@]\!\}$.
3. One element belongs to T_1 and the other one to T_2. Since S_2 is a sub-tree of S_1 with root a the only possibility is that ${}_{c}^{d} \in T_2$, ${}_{d}^{e} \in T_1$ and $d = a$. Moreover, since b is the father of a in S_1, it means that $e = b$. It is immediate to check that $c^{a^@} \in {T'_2}^\diamond$, so that $c^{a^b} \in {T'_2}^\diamond\{\![a^b/a^@]\!\}$.

We now show the converse $T'^\diamond \supseteq {T'_1}^\diamond \cup {T'_2}^\diamond\{\![a^b/a^@]\!\}$. Let us consider a generic element $c^{d^e} \in {T'_1}^\diamond \cup {T'_2}^\diamond\{\![a^b/a^@]\!\}$. There are two possibilities:

1. If $c^{d^e} \in {T'_1}^\diamond$, then both c^d, $d^e \in T_1$. It follows that both c^d, $d^e \in T_1 \cup T_2$, and thus $c^{d^e} \in T'^\diamond$.
2. If $c^{d^e} \in {T'_2}^\diamond\{\![a^b/a^@]\!\}$, then either $c^{d^e} \in {T'_2}^\diamond$ or $d = a$, $e = b$ and $c^{a^@} \in {T'_2}^\diamond$. The former case is analogous to 1. above. In the latter case we observe that $c^a \in T_2$. Since $a^b \in T_1$, we have $c^a, a^b \in T_1 \cup T_2$, and thus $c^{a^b} \in T'^\diamond$.

A similar argument applies also to the configuration. Hence, we have

$$(T'^\diamond, C'^\diamond) = ({T'_1}^\diamond, {C'_1}^\diamond) \cup ({T'_2}^\diamond, {C'_2}^\diamond)\{\![a^b/a^@]\!\}.$$

Therefore, we have also

$$(T'^\diamond, C'^\diamond)\rho^\diamond_{S_1\cup S_2}\eta^\diamond = ({T'_1}^\diamond, {C'_1}^\diamond)\rho^\diamond_{S_1\cup S_2}\eta^\diamond \ \cup\ ({T'_2}^\diamond, {C'_2}^\diamond)\{\![a^b/a^@]\!\}\rho^\diamond_{S_1\cup S_2}\eta^\diamond.$$

Using $a^\diamond = a\rho^\diamond_{S_1\cup S_2}\eta^\diamond$ and $b^\diamond = b\rho^\diamond_{S_1\cup S_2}\eta^\diamond$, we obtain

$$(T'^\diamond, C'^\diamond)\rho^\diamond_{S_1\cup S_2}\eta^\diamond = ({T'_1}^\diamond, {C'_1}^\diamond)\rho^\diamond_{S_1\cup s_2}\eta^\diamond \ \cup\ ({T'_2}^\diamond, {C'_2}^\diamond)\rho^\diamond_{S_1\cup S_2}\eta^\diamond\{\![a^{\diamond b^\diamond}/a^{\diamond @}]\!\}.$$

Now we observe that the equality is preserved, when the renamings $\rho^\diamond_{S_i}$ are used for $i \in \{1,2\}$ in place of $\rho^\diamond_{S_1\cup S_2}$ and $\cup$ is replaced by $\cup^\diamond$. This because $\cup^\diamond$ modifies the multiplicity counting the number of the occurrences of the union. Therefore, we conclude

$$(T'^\diamond, C'^\diamond)\rho^\diamond_{S_1\cup S_2}\eta^\diamond = ({T'_1}^\diamond, {C'_1}^\diamond)\rho^\diamond_{S_1}\eta^\diamond \ \cup^\diamond\ ({T'_2}^\diamond, {C'_2}^\diamond)\rho^\diamond_{S_2}\eta^\diamond\{\![a^{\diamond b^\diamond}/a^{\diamond @}]\!\}.$$

$\square$

The following proposition shows the safeness of the abstract normalisation function $\delta^\diamond$. Notice that a is the root of $\delta\ {}^a P$ so that the abstraction $\alpha^\diamond$ assigns @ as father of $a^\diamond$. It is therefore necessary to replace @ with $b^\diamond$.

Proposition B.7 *Let P be a well-labeled process and $a \in \mathcal{A}$ such that a is fresh for P. We have*

$$\alpha^\diamond(\delta\ {}^a P)\{\![a^{\diamond b^\diamond}/a^{\diamond @}]\!\}\subseteq^\diamond \delta^\diamond\ {}^{a^{\diamond b^\diamond}} P^\diamond.$$

Proof: The proof proceeds by induction on the structure of P using the definition of $\delta^\diamond$ (Table 6). We show the most interesting cases.

- Assume that $P = c[P_1]$. We have

$$\delta\ {}^a P = (\{c^a\}, \emptyset)\ \cup \delta\ {}^c P_1.$$

  By Proposition A.13, $\delta\ {}^a P$ is well-labeled state. Moreover, we observe that $\delta\ {}^c P_1$ is a sub-tree of $(\{c^a\}, \emptyset)$ with root c. Thus, by Lemma B.6 we have

$$\alpha^\diamond(\delta\ {}^a P) = \alpha^\diamond((\{c^a\}, \emptyset)) \cup^\diamond \alpha^\diamond(\delta\ {}^c P_1)\{\![c^{\diamond a^\diamond}/c^{\diamond @}]\!\}.$$

  By induction hypothesis we have

$$\alpha^\diamond(\delta\ {}^c P_1)\{\![c^{\diamond a^\diamond}/c^{\diamond @}]\!\}\subseteq^\diamond \delta^\diamond\ {}^{c^{\diamond a^\diamond}} P_1^\diamond.$$

  Moreover, by definition of $\alpha^\diamond$ we have $\alpha^\diamond((\{c^a\}, \emptyset)) = (\{c^{\diamond a^{\diamond @}}\}, \emptyset)$.
  Therefore, we have

$$\alpha^\diamond(\delta\ {}^a P)\subseteq^\diamond(\{c^{\diamond a^{\diamond @}}\}, \emptyset)\cup^\diamond \delta^\diamond\ {}^{c^{\diamond a^\diamond}} P_1^\diamond.$$

  We now observe that the replacement $\{\![a^{\diamond b^\diamond}/a^{\diamond @}]\!\}$ cannot affect $\delta^\diamond\ {}^{c^{\diamond a^\diamond}} P_1^\diamond$ because the abstract topology of a single state is a tree. Therefore, we have

$$\alpha^\diamond(\delta\ {}^a P)\{\![a^{\diamond b^\diamond}/a^{\diamond @}]\!\}\subseteq^\diamond(\{c^{\diamond a^{\diamond @}}\}, \emptyset)\{\![a^{\diamond b^\diamond}/a^{\diamond @}]\!\}\cup^\diamond \delta^\diamond\ {}^{c^{\diamond a^\diamond}} P_1^\diamond.$$

  We conclude because by definition of $\delta^\diamond$ we have

$$\delta^\diamond\ {}^{a^{\diamond b^\diamond}} P^\diamond = (\{c^{\diamond a^{\diamond b^\diamond}}\}, \emptyset)\cup^\diamond \delta^\diamond\ {}^{c^{\diamond a^\diamond}} P_1^\diamond.$$

- Assume that $P = (\nu n_\lambda P_1)$. We have

$$\delta\ {}^a P = \delta\ {}^a(P_1[\hat{n}_i/n])$$

  where $\hat{n}_i = H_{\mathcal{L}_I}(\lambda)$ and $\lambda = \ell_i$.
  Since P is well-labeled, then $\lambda \notin \Lambda(P_1)$. Therefore, $P_1[\hat{n}_i/n]$ is well-labeled, and by induction hypothesis we have

$$\alpha^\diamond(\delta\ {}^a(P_1[\hat{n}_i/n]))\{\![a^{\diamond b^\diamond}/a^{\diamond @}]\!\}\subseteq^\diamond \delta^\diamond\ {}^{a^{\diamond b^\diamond}} \alpha^\diamond(P_1[\hat{n}_i/n]).$$

  Let $P^\diamond = \alpha^\diamond(P) = (\nu n_{\lambda^\diamond})\ P_1^\diamond$. By definition of $\alpha^\diamond$ we have $\alpha^\diamond(P_1[\hat{n}_i/n]) = P_1^\diamond[\hat{n}/n]$. Now we use $H_{\mathcal{L}^\diamond}(\lambda^\diamond) = \hat{n}$ and we obtain by definition of $\delta^\diamond$

$$\delta^\diamond\ {}^{a^{\diamond b^\diamond}} P^\diamond = \delta^\diamond\ {}^{a^{\diamond b^\diamond}} P_1{}^\diamond[\hat{n}/n].$$

$\square$

The following lemma is the core of the proof of safeness; it states the agreement between concrete and abstract transitions.

Lemma B.8 *Let $S, S' \in \mathcal{S}$ be well-labeled states. For any $S \mapsto S'$ there exists an abstract state $S'^{\diamond}$, such that $\alpha^{\diamond}(S) \mapsto^{\diamond} S'^{\diamond}$ and $\alpha^{\diamond}(S') \subseteq^{\diamond} S'^{\diamond}$.*

Proof: The proof is by cases on the rule applied to obtain the transition $S \mapsto S'$. One of the rules **Bang**, **In**, **Out** and **Open** of Table 5 could have been applied. Assume that $S = (T, C)$, by definition of $\alpha^{\diamond}$ (Definition 5.5), we have $\alpha^{\diamond}(S) = (T^{\diamond}, C^{\diamond}) = (T'^{\diamond}, C'^{\diamond})\rho^{\diamond}_{S}\eta^{\diamond}$ where

$$T'^{\diamond} = \{\ {}_{a}^{\,b^{c}} \mid {}_{a}^{\,b},\ {}_{b}^{\,c} \in T\}$$

$$C'^{\diamond} = \{\ {}^{a^{b}}P \mid {}_{a}^{\,b} \in T,\ {}^{a}P \in C\}.$$

As usual we use $a^{\diamond}$ to denote the abstract version of a, that is $a\rho^{\diamond}_{S}\eta^{\diamond}$. Similarly, for the other ambients and processes.

Bang It means that ${}^{a}!P \in C$ and that

$$S' = S \cup \delta\ {}^{a}new_{S}(P).$$

By definition of $\alpha^{\diamond}$ we have ${}^{a^{\diamond}b^{\diamond}}!P^{\diamond} \in C^{\diamond}$, where b is the father of a in S, i.e. either ${}_{a}^{\,b} \in T$, or a is the root of T and $b = @$. Hence, by applying rule **Bang**$^{\diamond}$, we obtain a transition $\alpha^{\diamond}(S) \mapsto^{\diamond} S'^{\diamond}$ where

$$S'^{\diamond} = \alpha^{\diamond}(S) \cup^{\diamond} \delta^{\diamond}\ {}^{a^{\diamond}b^{\diamond}}new_{\omega}(P^{\diamond}).$$

It remains to show that $\alpha^{\diamond}(S') \subseteq^{\diamond} S'^{\diamond}$, that is

$$\alpha^{\diamond}(S \cup \delta\ {}^{a}new_{S}(P)) \subseteq^{\diamond} \alpha^{\diamond}(S) \cup^{\diamond} \delta^{\diamond}\ {}^{a^{\diamond}b^{\diamond}}new_{\omega}(P^{\diamond}).$$

We observe that $\delta\ {}^{a}new_{S}(P)$ is a sub-tree of S with root a. Hence, by Lemma B.6, we have

$$\alpha^{\diamond}(S \cup \delta\ {}^{a}new_{S}(P)) = \alpha^{\diamond}(S) \cup^{\diamond} \alpha^{\diamond}(\delta\ {}^{a}new_{S}(P))\{\![a^{\diamond b^{\diamond}}/a^{\diamond @}]\!\}.$$

By Proposition B.7 we have also

$$\alpha^{\diamond}(\delta\ {}^{a}new_{S}(P))\{\![a^{\diamond b^{\diamond}}/a^{\diamond @}]\!\} \subseteq^{\diamond} \delta^{\diamond}\ {}^{a^{\diamond}b^{\diamond}}\alpha^{\diamond}(new_{S}(P)).$$

We now observe that the function new_{ω} gives multiplicity ω to any label of $P^{\diamond}$. It means that

$$\delta^{\diamond}\ {}^{a^{\diamond}b^{\diamond}}\alpha^{\diamond}(new_{S}(P)) \subseteq^{\diamond} \delta^{\diamond}\ {}^{a^{\diamond}b^{\diamond}}new_{\omega}(P^{\diamond}).$$

We therefore conclude

$$\alpha^{\diamond}(\delta\ {}^{a}new_{S}(P))\{\![a^{\diamond b^{\diamond}}/a^{\diamond @}]\!\} \subseteq^{\diamond} \delta^{\diamond}\ {}^{a^{\diamond}b^{\diamond}}new_{\omega}(P^{\diamond}).$$

In It means that ${}_{a}{}^{b},\ {}_{m_\mu}{}^{b} \in T$ and $t = {}^{a}\mathtt{in}\,m_\gamma.P \in C$, where $a \neq m_\mu$ and $a \neq @$. Moreover,

$$S' = \delta\ {}^{a}P\ \cup\ ((T \setminus \{\,{}_{a}{}^{b}\}) \cup \{\,{}_{a}{}^{m_\mu}\}, C \setminus \{t\}).$$

By definition of $\alpha^\diamond$ we have ${}^{{a^\diamond}^{b^\diamond}}\mathtt{in}\,m_{\gamma^\diamond}.P^\diamond \in C^\diamond$, since b is the father of a (${}_{a}{}^{b} \in T$). Moreover, it is immediate to check that there exists $c^\diamond$ such that ${}_{a^\diamond}{}^{{b^\diamond}^{c^\diamond}},\ {}_{m_{\mu^\diamond}}{}^{{b^\diamond}^{c^\diamond}} \in T^\diamond$. Notice that, since $a \neq @$, either c is the father of b in T, or $b = @$ and $c^\diamond = \top$, or b is the root of T and $c^\diamond = @$.

We now observe that the side condition of rule **In**$^\diamond$ is satisfied (if $a^\diamond = m_{\ell_1}$ then $\mu^\diamond \neq \ell_1$). Since $a \neq m_\mu$, there are two cases: either $a = k_\lambda$ or $a = m_\lambda$ with $\lambda \neq \mu$. In the former case the side condition is immediately satisfied. In the latter case it depends on whether λ and μ differ in the indexes only. In particular, when $\lambda = \ell_j$ and $\mu = \ell_h$ for indexes j, h, such that $j \neq h$, the side condition is satisfied, because $\lambda^\diamond = \mu^\diamond = \ell_\omega$ by definition of the abstraction. By applying rule **In**$^\diamond$, we obtain a transition $\alpha^\diamond(S) {\mapsto}^\diamond S'^\diamond$ where

$$S'^\diamond = \delta^\diamond\ {}_{a^\diamond}{}^{m_{\mu^\diamond}} P^\diamond \cup^\diamond S_2^\diamond$$

$$S_2^\diamond = \alpha^\diamond(S) \cup^\diamond (T^\diamond \cup^\diamond \{\,{}_{a^\diamond}{}^{{m_{\mu^\diamond}}^{b^\diamond}}\}, C^\diamond \setminus^\diamond \{t^\diamond\})\{\![a^\diamond{}^{m_{\mu^\diamond}} / a^\diamond{}^{b^\diamond}]\!\}$$

It remains to show that $\alpha^\diamond(S') \subseteq^\diamond S'^\diamond$, that is

$$\alpha^\diamond(\delta\ {}^{a}P\ \cup\ ((T \setminus \{\,{}_{a}{}^{b}\}) \cup \{\,{}_{a}{}^{m_\mu}\}, C \setminus \{t\})) \subseteq^\diamond S'^\diamond.$$

We observe that $\delta\ {}^{a}P$ is a sub-tree of S' with root a and that the father of a in S' is m_μ. Hence, by Lemma B.6, we have

$$\alpha^\diamond(S') = \alpha^\diamond(\delta\ {}^{a}P)\{\![a^\diamond{}^{{m_\mu}^\diamond} / a^\diamond{}^{@}]\!\} \cup^\diamond S_1^\diamond$$

$$S_1^\diamond = \alpha^\diamond((T \setminus \{\,{}_{a}{}^{b}\}) \cup \{\,{}_{a}{}^{m_\mu}\}, C \setminus \{t\}).$$

By Proposition B.7 we have also

$$\alpha^\diamond(\delta\ {}^{a}P)\ \{\![a^\diamond{}^{{m_\mu}^\diamond} / a^\diamond{}^{@}]\!\} \subseteq^\diamond \delta^\diamond\ {}^{a^\diamond{}^{{m_\mu}^\diamond}} P^\diamond.$$

Hence, to conclude it is enough to show that ${S_1}^\diamond \subseteq^\diamond {S_2}^\diamond$, that is

$$\alpha^\diamond((T \setminus \{\,{}_{a}{}^{b}\}) \cup \{\,{}_{a}{}^{m_\mu}\}, C \setminus \{t\}) \subseteq^\diamond \alpha^\diamond(S) \cup^\diamond (T^\diamond \cup^\diamond \{\,{}_{a^\diamond}{}^{{m_{\mu^\diamond}}^{b^\diamond}}\}, C^\diamond \setminus^\diamond \{t^\diamond\})\{\![a^\diamond{}^{m_{\mu^\diamond}} / a^\diamond{}^{b^\diamond}]\!\}.$$

In the following we assume that ${S_i}^\diamond = ({T_i}^\diamond, {C_i}^\diamond)$ for $i \in \{1, 2\}$ (we recall also that $\alpha^\diamond(S) = (T^\diamond, C^\diamond) = (T'^\diamond, C'^\diamond)\rho_S^\diamond \eta^\diamond$).

- We show $T_1^\diamond \subseteq^\diamond T_2^\diamond$. Let ${}_{d^\diamond}{}^{{e^\diamond}^{f^\diamond}} \in {T_1}^\diamond$, by definition of $\alpha^\diamond$ we have ${}_{d}{}^{e},\ {}_{e}{}^{f} \in (T \setminus \{\,{}_{a}{}^{b}\}) \cup \{\,{}_{a}{}^{m_\mu}\}$. There are several cases to consider depending on how the ambient a, whose father has changed, is involved.

  Assume that none of $d^\diamond$ and $e^\diamond$ is equal to $a^\diamond$. It is easy to check that ${}_{d}{}^{e},\ {}_{e}{}^{f} \in T$. Hence, we have ${}_{d^\diamond}{}^{{e^\diamond}^{f^\diamond}} \in T^\diamond$ and, consequently, also ${}_{d^\diamond}{}^{{e^\diamond}^{f^\diamond}} \in {T_2}^\diamond$.

Assume that $d^\diamond = a^\diamond$. Since ${}_{m_\mu}{}^{b}$, ${}_{a}{}^{m_\mu} \in (T \setminus \{ {}_{a}{}^{b}\}) \cup \{ {}_{a}{}^{m_\mu}\}$ then $e^\diamond = m_{\mu^\diamond}$ and $f^\diamond = b^\diamond$. We conclude because ${}_{a^\diamond}{}^{{m_\mu^\diamond}^{b^\diamond}} \in {T_2}^\diamond$.
Assume that $e^\diamond = a^\diamond$. It means that $f^\diamond = m_{\mu^\diamond}$ as ${}_{a}{}^{m_\mu} \in (T\setminus\{ {}_{a}{}^{b}\})\cup\{ {}_{a}{}^{m_\mu}\}$. Therefore, we have ${}_{d}{}^{a}$, ${}_{a}{}^{b} \in T$ and ${}_{d^\diamond}{}^{{a^\diamond}^{b^\diamond}} \in T^\diamond$. Moreover , we have ${}_{d^\diamond}{}^{{a^\diamond}^{m_{\mu^\diamond}}} \in T^\diamond\{\![a^{\diamond m_\mu{}^\diamond}/a^{\diamond b^\diamond}]\!\}$ and, consequently, ${}_{d^\diamond}{}^{{a^\diamond}^{m_{\mu^\diamond}}} \in {T_2}^\diamond$.

- We show $C_1^\diamond \subseteq^\diamond {C_2}^\diamond$ by considering a generic element ${}^{{d^\diamond}^{e^\diamond}}Q \in C_1^\diamond$. The proof is similar to the one shown for the topology; the only interesting case is when the process Q is local to a, that is $d^\diamond = a^\diamond$ and $e^\diamond = {m_\mu}^\diamond$. By definition of $\alpha^\diamond$ it means that ${}^{a}Q \in C\setminus\{t\}$ and $a^{m_\mu} \in (T \setminus \{ {}_{a}{}^{b}\}) \cup \{ {}_{a}{}^{m_\mu}\}$, and consequently ${}^{a}Q \in C$ and $a^b \in T$. By definition of $\alpha^\diamond$ we obtain ${}^{{a^\diamond}^{b^\diamond}}Q^\diamond \in C^\diamond$. Now, we use the definition of $\setminus^\diamond$; there are two cases depending on whether the label γ is either ℓ'_1 or ℓ'_ω.
  When $\gamma = \ell'_\omega$ we have $C^\diamond\setminus^\diamond\{t^\diamond\} = C^\diamond$. Since ${}^{{a^\diamond}^{b^\diamond}}Q^\diamond \in C^\diamond$ then we conclude ${}^{{a^\diamond}^{m_\mu{}^\diamond}}Q^\diamond \in C^\diamond\setminus^\diamond\{t^\diamond\}\{\![a^{\diamond m_\mu{}^\diamond}/a^{\diamond b^\diamond}]\!\}$.
  When $\gamma = \ell'_1$ we have $C^\diamond\setminus^\diamond\{t^\diamond\} = C^\diamond \setminus \{t^\diamond\}$. We observe that it cannot be the case that $Q^\diamond = t^\diamond$, as $\gamma = \ell_1$ shows that there is only object with label γ. Therefore, we have ${}^{{a^\diamond}^{b^\diamond}}Q^\diamond \in C^\diamond\setminus^\diamond\{t^\diamond\}$. We then conclude as before.

Out Similar to the case of rule **In**$^\diamond$ above.

Open It means that ${}_{m_\mu}{}^{a} \in T$ and $t = {}^{a}\mathtt{open}\, m_\gamma . P \in C$, where $a \neq m_\mu$. Moreover,

$$S' = \delta\ {}^{a}P \ \cup\ ((T \setminus \{ {}_{m_\mu}{}^{a}\}), (C \setminus \{t\}))\{\![a/m_\mu]\!\}$$

By definition of $\alpha^\diamond$ we have ${}^{{a^\diamond}^{b^\diamond}}\mathtt{open}\, m_{\lambda^\diamond} . P \in C^\diamond$, where b is the father of a in T, i.e. either ${}_{a}{}^{b} \in T$ or a is the root of T and $b = @$. Moreover, since ${}_{m_\mu}{}^{a} \in T$ we have also ${}_{m_\mu{}^\diamond}{}^{{a^\diamond}^{b^\diamond}} \in T^\diamond$. We observe that the side condition of rule **Open**$^\diamond$ is satisfied since $a \neq m_\mu$ (by applying a reasoning similar to that for **In**). Hence, by applying rule **Open**$^\diamond$, we obtain a transition $\alpha^\diamond(S) \mapsto^\diamond S'^\diamond$ where

$$S'^\diamond = \delta^\diamond\ {}^{{a^\diamond}^{b^\diamond}}P^\diamond \cup^\diamond \alpha^\diamond(S) \cup^\diamond \alpha^\diamond(S)\{\![a^{\diamond b^\diamond}/{m_\mu}^{\diamond a^\diamond}]\!\}\{\![c^{a^\diamond}/c^{m_\mu{}^\diamond}]\!\}$$

It remains to show that $\alpha^\diamond(S') \subseteq^\diamond S'^\diamond$, that is

$$\alpha^\diamond(\delta\ {}^{a}P \ \cup\ ((T \setminus \{ {}_{m_\mu}{}^{a}\}), (C \setminus \{t\}))\{\![a/m_\mu]\!\}) \subseteq^\diamond S'^\diamond.$$

We observe that $\delta\ {}^{a}P$ is a sub-tree of S with root a. Hence, by Lemma B.6, we have

$$\alpha^\diamond(S') = \alpha^\diamond(\delta\ {}^{a}P)\{\![a^{\diamond b^\diamond}/a^{\diamond @}]\!\} \cup^\diamond \alpha^\diamond(((T \setminus \{ {}_{m_\mu}{}^{a}\}), (C \setminus \{t\}))\{\![a/m_\mu]\!\}).$$

By Proposition B.7 we have also

$$\alpha^\diamond(\delta\ {}^{a}P)\ \{\![a^{\diamond b}/a^{\diamond @}]\!\} \subseteq^\diamond \delta^\diamond\ {}^{{a^\diamond}^{b^\diamond}}P^\diamond.$$

Therefore, to conclude it is enough to show that

$$\alpha^\diamond(((T \setminus \{ {}_{m_\mu}{}^{a}\}), (C \setminus \{t\}))\{\![a/m_\mu]\!\}) \subseteq^\diamond \alpha^\diamond(S) \cup^\diamond \alpha^\diamond(S)\{\![a^{\diamond b^\diamond}/{m_\mu}^{\diamond a^\diamond}]\!\}\{\![c^{a^\diamond}/c^{m_\mu{}^\diamond}]\!\}.$$

This can be shown following the reasoning used for the similar inclusion in rule **In** above. It is worth giving some details only about the substitutions. The substitution $\{\![a^{\diamond b^\diamond}/m_\mu{}^{\diamond a^\diamond}]\!\}$ guarantees that the opening ambient a acquires any ambient and process local to m_μ. Similarly, the substitution $\{\![c^{a^\diamond}/c^{m_\mu{}^\diamond}]\!\}$ guarantees that the removal of m_μ is propagated also to the processes and ambients local to an ambient, which is a son of m_μ.

□

We can now prove the main result, that is Lemma 5.8. We recall its assertion for clarity: *Let* $S_2 \in \mathcal{S}$ *and* $S^\natural \in \mathcal{S}^\natural$. *We have*

$$\alpha^\diamond(\Psi_{S_2}(S^\natural)) \subseteq^\diamond \Psi^\diamond_{\alpha^\diamond(S_2)}(\alpha^\diamond(S^\natural)).$$

Proof: [of Lemma 5.8] We first notice that, by definition of $\sim$, when $S_1 \sim S_2$ we have $\alpha^\diamond(S_1) = \alpha^\diamond(S_2)$. Moreover, for any $S_1 \mapsto S_1'$ we have $S_2 \mapsto S_2'$ such that $S_1' \sim S_2'$. This observation permits us to simplify the proof by using, with an abuse of notation, $S \in S^\natural$ in place of $[S] \in S^\natural$. By definition of Ψ_{S_2} (Definition 4.9) we therefore have

$$\Psi_{S_2}(S^\natural) = \{[S_2]\} \cup \bigcup_{S \in \{S_3 \mid S_1 \mapsto S_3,\ S_1 \in S^\natural\}} \{[S]\}.$$

Thus, by continuity of $\alpha^\diamond$ (Proposition B.3) and using $\alpha^\diamond(\{[S_2]\}) = \alpha^\diamond(S_2)$ and $\alpha^\diamond(\{[S]\}) = \alpha^\diamond(S)$, we obtain

$$\alpha^\diamond(\Psi_{S_2}(S^\natural)) = \alpha^\diamond(S_2) \cup^\diamond \bigcup^\diamond_{S \in \{S_3 \mid S_1 \mapsto S_3,\ S_1 \in S^\natural\}} \alpha^\diamond(S).$$

By Lemma B.8 we have that, for each $S_1 \in S^\natural$ and for each $S_1 \mapsto S_3$, there exists $\alpha^\diamond(S_1) \mapsto^\diamond S_3^\diamond$ such that $\alpha^\diamond(S_3) \subseteq^\diamond S_3^\diamond$. Since $\{S_1\} \subseteq S^\natural$, then by monotonicity of $\alpha^\diamond$ (Proposition B.3) we have $\alpha^\diamond(\{S_1\}) = \alpha^\diamond(S_1) \subseteq^\diamond \alpha^\diamond(S^\natural)$. Hence, by Lemma B.4, we have also $\alpha^\diamond(S^\natural) \mapsto^\diamond S_4^\diamond$ such that $\alpha^\diamond(S_3) \subseteq^\diamond S_3^\diamond \subseteq^\diamond S_4^\diamond$.
We conclude, because by Definition of $\Psi^\diamond_{\alpha^\diamond(S_2)}$ (Definition 5.7), we have

$$\Psi^\diamond_{\alpha^\diamond(S_2)}(\alpha^\diamond(S^\natural)) = \alpha^\diamond(S_2) \cup^\diamond \bigcup^\diamond_{S^\diamond \in \{S_3^\diamond \mid \alpha^\diamond(S^\natural) \mapsto^\diamond S_3^\diamond\}} S^\diamond.$$

□

B.2 Second abstraction

We first show that the pair of functions $(\alpha^\circ, \gamma^\circ)$ forms a Galois connection between $\langle \mathcal{S}^\diamond, \subseteq^\diamond \rangle$ and $\langle \mathcal{S}^\circ, \subseteq \rangle$ (Theorem 6.4).

Proposition B.9 *The abstract domain* $\langle \mathcal{S}^\circ, \subseteq \rangle$ *is a* complete lattice.

Proof: Straightforward by Proposition B.1. □

The following proposition states the basic properties of the concretization and abstraction functions.

Proposition B.10 *Function* $\alpha^\circ : \langle \mathcal{S}^\diamond, \subseteq^\diamond \rangle \to \langle \mathcal{S}^\circ, \subseteq \rangle$ *is* monotonic *and* continuous *and function* $\gamma^\circ : \langle \mathcal{S}^\circ, \subseteq \rangle \to \langle \mathcal{S}^\diamond, \subseteq^\diamond \rangle$ *is* monotonic.

Proof: Trivial by Definition 6.3. □

The properties stated above are enough to prove Theorem 6.4.

Proof: [of Theorem 6.4] We show that $(\alpha^\circ, \gamma^\circ)$ is a Galois connection (see Definition 2.1). By Propositions B.2 and B.9 both abstract domains are complete lattices. Also, by Proposition B.10 both α° and γ° are monotonic. Hence, it remains two show that, for $S^\circ \in \mathcal{S}^\circ$ and $S^\diamond \in \mathcal{S}^\diamond$, we have

$$S^\diamond \subseteq^\diamond \gamma^\circ(\alpha^\circ(S^\diamond))$$

$$\alpha^\circ(\gamma^\circ(S^\circ)) \subseteq S^\circ$$

Both assertions follow straightforwardly from Definition 6.3. We have $S^\diamond \subseteq^\diamond \gamma^\circ(\alpha^\circ(S^\diamond))$ since, by definition of γ° and α°,

$$\gamma^\circ(\alpha^\circ(S^\diamond)) = \bigcup^\diamond \{S'^\diamond \mid \alpha^\circ(S'^\diamond) \subseteq \alpha^\circ(S^\diamond)\}.$$

Moreover, by definition of γ° and α° and by continuity of α° (Proposition B.10) , we have

$$\alpha^\circ(\gamma^\circ(S^\circ)) = \alpha^\circ(\bigcup^\diamond \{S^\diamond \mid \alpha^\circ(S^\diamond) \subseteq S^\circ\}) = \bigcup \alpha^\circ(\{S^\diamond \mid \alpha^\circ(S^\diamond) \subseteq S^\circ\}).$$

By definition of least upper bound on a complete lattice we conclude therefore

$$\bigcup \alpha^\circ(\{S^\diamond \mid \alpha^\circ(S^\diamond) \subseteq S^\circ\}) \subseteq S^\circ.$$

□

We now show the safeness of the second abstraction (Lemma 6.6). The proof uses some auxiliary lemmata similar to those shown for the first abstraction.

Lemma B.11 *Let* $S_1^\circ, S_2^\circ \in \mathcal{S}^\circ$ *be well-labeled abstract states such that* $S_1^\circ \subseteq S_2^\circ$*. if* $S_1^\circ \mapsto^\circ S'^\circ_1$*, then there exists a transition* $S_2^\circ \mapsto^\circ S'^\circ_2$*, such that* $S'^\circ_1 \subseteq^\circ S'^\circ_2$*.*

Proof: The proof is straightforward by cases on the rules of Table 8.

□

Lemma B.12 *Let* $S^\circ \in \mathcal{S}^\circ$*. The function* $\Psi^\circ_{S^\circ} : \langle \mathcal{S}^\circ, \subseteq \rangle \to \langle \mathcal{S}^\circ, \subseteq \rangle$ *is monotonic.*

Proof: This follows from Lemma B.11 using Definition 6.5.

□

To simplify the notation we use the following convention: a° denotes the abstract version of a, that is $a\rho^\circ$ where ρ° is the renaming which forgets multiplicities (i.e. $\rho^\circ(\ell_1) = \rho^\circ(\ell_\omega) = \ell$.) Similarly for processes P° is the abstract version of P.

Proposition B.13 *Let P be a well-labeled abstract process. We have*

$$\alpha^\circ(\delta^\diamond\ {}_{a}^{b}P) = \delta^\circ\ {}^{a^\circ}P^\circ.$$

Proof: The proof is easy proceeding by induction on the structure of P and using the definition of α° (Definition 6.3). We recall that

$$\alpha^\circ((T^\diamond, C^\diamond)) = (\{\ {}_{a}^{b} \mid\ {}_{a}^{b^c} \in T^\diamond\}, \{\ {}^{a}P \mid\ \ {}_{a}^{b}P \in C^\diamond\})\rho^\circ.$$

Since function α° removes the partial topology, the information that b is father of a is lost.

$\square$

Lemma B.14 *Let $S^\diamond, S'^\diamond \in \mathcal{S}^\diamond$ be well-labeled abstract states. For any $S^\diamond \mapsto^\diamond S'^\diamond$ there exists an abstract state S'°, such that $\alpha^\circ(S^\diamond) \mapsto^\circ S'^\circ$ and $\alpha^\circ(S'^\diamond) \subseteq S'^\circ$.*

Proof: The proof is by cases on the rule applied to obtain the transition $S^\diamond \mapsto^\diamond S'^\diamond$. One of the rules **Bang**$^\diamond$, **In**$^\diamond$, **Out**$^\diamond$ and **Open**$^\diamond$ of Table 7 could have been applied. Let $S^\diamond = (T^\diamond, C^\diamond)$ and $\alpha^\circ(S^\diamond) = (T^\circ, C^\circ)$. We recall that, by definition of α° (Definition 6.3), we have

$$\alpha^\circ(S^\diamond) = (\{\ {}_{a}^{b} \mid\ {}_{a}^{b^c} \in T^\diamond\}, \{\ {}^{a}P \mid\ \ {}_{a}^{b}P \in C^\diamond\})\rho^\circ$$

Bang$^\diamond$ It means that ${}_{a}^{b}!P \in C^\diamond$ and that

$$S'^\diamond = S^\diamond \cup^\diamond \delta^\diamond\ {}_{a}^{b}new_\omega(P).$$

By definition of α° we derive that ${}^{a^\circ}!P^\circ \in C^\circ$. Hence, by applying rule **Bang**$^\circ$ of Table 8, we obtain a transition $\alpha^\circ(S^\diamond) \mapsto^\circ S'^\circ$ where

$$S'^\circ = \alpha^\circ(S^\diamond) \cup \delta^\circ\ {}^{a^\circ}P^\circ.$$

It remains to show that $\alpha^\circ(S'^\diamond) \subseteq S'^\circ$. By continuity of α° (Proposition B.10) we have

$$\alpha^\circ(S'^\diamond) = \alpha^\circ(S^\diamond)\ \cup\ \alpha^\circ(\delta^\diamond\ {}_{a}^{b}new_\omega(P)).$$

We notice that, since the abstraction α° forgets any multiplicity, we have

$$\alpha^\circ(\delta^\diamond\ {}_{a}^{b}new_\omega(P)) = \alpha^\circ(\delta^\diamond\ {}_{a}^{b}P).$$

We conclude, because by Proposition B.13 we have

$$\alpha^\circ(\delta^\diamond\ {}_{a}^{b}P) = \delta^\circ\ {}^{a^\circ}P^\circ.$$

In$^\diamond$ It means that ${}_{a}^{b^c}$, ${}_{m_\mu}^{b^c} \in T^\diamond$ and ${}_{a}^{b}\texttt{in}\, m_\lambda.P \in C^\diamond$, and that

$$S'^\diamond = S^\diamond \cup^\diamond \delta^\diamond\ {}_{a}^{m_\mu}P \cup^\diamond (T^\diamond\ \cup\ \{\ {}_{a}^{m_\mu^b}\}, C^\diamond \setminus^\diamond \{c\})[\![a^{m_\mu}/a^b]\!].$$

By definition of α° we have that ${}_{a^\circ}^{b^\circ}$, ${}_{m_{\mu^\circ}}^{b^\circ} \in T^\circ$ and ${}^{a^\circ}\texttt{in}\, m_{\lambda^\circ}.P^\circ \in C^\circ$. Hence, by applying rule **In**$^\circ$ of Table 8, we obtain a transition $\alpha^\circ(S^\diamond) \mapsto^\circ S'^\circ$ where

$$S'^\circ = \alpha^\circ(S^\diamond)\ \cup\ \delta^\circ\ {}^{a^\circ}P^\circ\ \cup\ (\{\ {}_{a^\circ}^{m_{\mu^\circ}}\}, \emptyset).$$

It remains to show that $\alpha^\circ(S'^\diamond) \subseteq S'^\circ$. By continuity of α° (Proposition B.10) we have

$$\alpha^\circ(S'^\diamond) = \alpha^\circ(S^\diamond) \cup^\diamond \alpha^\circ(\delta^\diamond \ {}_{a}{}^{m_\mu} P) \cup^\diamond \alpha^\circ((T^\diamond \ \cup \ \{ \ {}_{a}{}^{m_\mu^b}\}, C^\diamond \backslash^\diamond \{c\})\{\![a^{m_\mu}/a^b]\!\}).$$

By Proposition B.13 we have that

$$\alpha^\circ(\delta^\diamond \ {}_{a}{}^{m_\mu} P) \subseteq \delta^\circ \ {}^{a^\circ} P^\circ.$$

We observe that

$$\alpha^\circ((T^\diamond \ \cup \ \{ \ {}_{a}{}^{m_\mu^b}\}, C^\diamond \backslash^\diamond \{c\})\{\![a^{m_\mu}/a^b]\!\}) = \alpha^\circ((T^\diamond \ \cup \ \{ \ {}_{a}{}^{m_\mu^b}\}, C^\diamond \backslash^\diamond \{c\})$$

In fact, the operation of replacement only affects the partial topology which is removed by the abstraction α°. Furthermore, we have also using the continuity of α° and the definition of $\backslash^\diamond$

$$\alpha^\circ((T^\diamond \cup^\diamond \{ \ {}_{a}{}^{m_\mu^b}\}, C^\diamond \backslash^\diamond \{c\}) \subseteq \alpha^\circ((T^\diamond \cup \{ \ {}_{a}{}^{m_\mu^b}\}, C^\diamond) = \alpha^\circ(S^\diamond) \ \cup \ \alpha^\circ((\ {}_{a}{}^{m_\mu^b}, \emptyset)).$$

Since $\alpha^\circ((\ {}_{a}{}^{m_\mu^b}, \emptyset)) = (\{ \ {}_{a^\circ}{}^{m_{\mu^\circ}}\}, \emptyset)$ we conclude that

$$\alpha^\circ((T^\diamond \ \cup \ \{ \ {}_{a}{}^{m_\mu^b}\}, C^\diamond \backslash^\diamond \{c\})\{\![a^{m_\mu}/a^b]\!\}) \subseteq \alpha^\circ(S^\diamond) \ \cup \ (\{ \ {}_{a^\circ}{}^{m_{\mu^\circ}}\}, \emptyset).$$

Out$^\diamond$ The proof is similar to that of rule **In**$^\diamond$ above.

Open$^\diamond$ It means that ${}_{m_\mu}{}^{a^b} \in T^\diamond$ and ${}_{a}{}^{b}$ `open` $m_{\lambda^\diamond}.P \in C^\diamond$ and that

$$S'^\diamond = S^\diamond \cup^\diamond \delta^\diamond \ {}_{a}{}^{b} P \cup^\diamond S^\diamond\{\![a^b/{m_\mu}^a]\!\}\{\![c^a/c^{m_\mu}]\!\}.$$

By definition of α° we have ${}_{m_{\mu^\circ}}{}^{a^\circ} \in T^\circ$ and ${}^{a^\circ}$ `open` $m_{\lambda^\circ}.P^\circ \in C^\circ$. Hence, by applying rule **Open**$^\circ$ of Table 8, we obtain a transition $\alpha(S^\diamond) \mapsto^\circ S'^\circ$ where

$$S'^\circ = \alpha^\circ(S^\diamond) \ \cup \ \delta^\circ \ {}^{a^\circ} P^\circ \ \cup \ \alpha^\circ(S^\diamond)\{\![a^\circ/m_{\ell'}]\!\}.$$

It remains to show that $\alpha^\circ(S'^\diamond) \subseteq S'^\circ$. By continuity of α° (Proposition B.10) we have

$$\alpha^\circ(S'^\diamond) = \alpha^\circ(S^\diamond) \ \cup \ \alpha^\circ(\delta^\diamond \ {}_{a}{}^{b} P) \ \cup \ \alpha^\circ(S^\diamond\{\![a^b/{m_\mu}^a]\!\}\{\![c^a/c^{m_\mu}]\!\}).$$

By Proposition B.13 we have that

$$\alpha^\circ(\delta^\diamond \ {}_{a}{}^{b} P) \subseteq \delta^\circ \ {}^{a^\circ} P^\circ.$$

We observe also that, since the abstraction α° forgets the partial topology, we have

$$\alpha^\circ(S^\diamond\{\![a^b/{m_\mu}^a]\!\}\{\![c^a/c^{m_\mu}]\!\}) = \alpha^\circ(S^\diamond\{\![a^b/{m_\mu}^a]\!\})$$

$$\alpha^\circ(S^\diamond\{\![a^b/{m_\mu}^a]\!\}) = \alpha^\circ(S^\diamond)\{\![a^\circ/m_{\ell'}]\!\}.$$

Hence, we conclude that

$$\alpha^\circ(S^\diamond\{\![a^b/{m_\mu}^a]\!\}\{\![c^a/c^{m_\mu}]\!\}) \subseteq \alpha^\circ(S^\diamond)\{\![a^\circ/m_{\ell'}]\!\}.$$

□

We can now show the proof of Lemma 6.6. We recall its assertion:
Let $S_1^\diamond, S_2^\diamond \in \mathcal{S}^\diamond$. *We have*

$$\alpha^\circ(\Psi^\diamond_{S_2^\diamond}(S_1^\diamond)) \subseteq \Psi^\circ_{\alpha^\circ(S_2^\diamond)}(\alpha^\circ(S_1^\diamond))$$

Proof: [of Lemma 6.6] The proof is analogous to that of Lemma 5.8 using Lemma B.14 and Lemma B.11. □

$P \gg P$	(Refl)
$P \gg Q,\ Q \gg R \Rightarrow P \gg R$	(Trans)
$P \mid Q \gg Q \mid P$	(Comm)
$(P \mid Q) \mid R \gg P \mid (Q \mid R)$ $P \mid (Q \mid R) \gg (P \mid Q) \mid R$	(Ass)
$P \gg Q \Rightarrow (\nu n_\lambda)P \gg (\nu n_\lambda)Q$	(Res)
$P \gg Q \Rightarrow P \mid R \gg Q \mid R$	(Par)
$P \gg Q \Rightarrow n_\lambda[P] \gg n_\lambda[Q]$	(Amb)
$n \neq m \Rightarrow (\nu n_\lambda)\ (\nu m_\mu)\ P \gg (\nu m_\mu)\ (\nu n_\lambda)\ P$	(Res-Com)
$n \notin fn(P) \Rightarrow \begin{array}{l} (\nu n_\lambda)\ (P \mid Q) \gg P \mid (\nu n_\lambda)\ Q \\ P \mid (\nu n_\lambda)\ Q \gg (\nu n_\lambda)\ (P \mid Q) \end{array}$	(Res-Par)
$n \neq m \Rightarrow \begin{array}{l} (\nu n_\lambda)\ m_\mu[P] \gg m_\mu[(\nu n_\lambda)\ P] \\ m_\mu[(\nu n_\lambda)\ P] \gg (\nu n_\lambda)\ m_\mu[P] \end{array}$	(Res-Amb)
$P \mid 0 \gg P \quad P \gg P \mid 0$	(Nil-Par)
$(\nu n_\lambda)0 \gg 0 \quad 0 \gg (\nu n_\lambda)0$	(Nil-Res)
$!P \gg new(P) \mid !P$	(Bang-Bang)

Table 10: The relation $\gg$